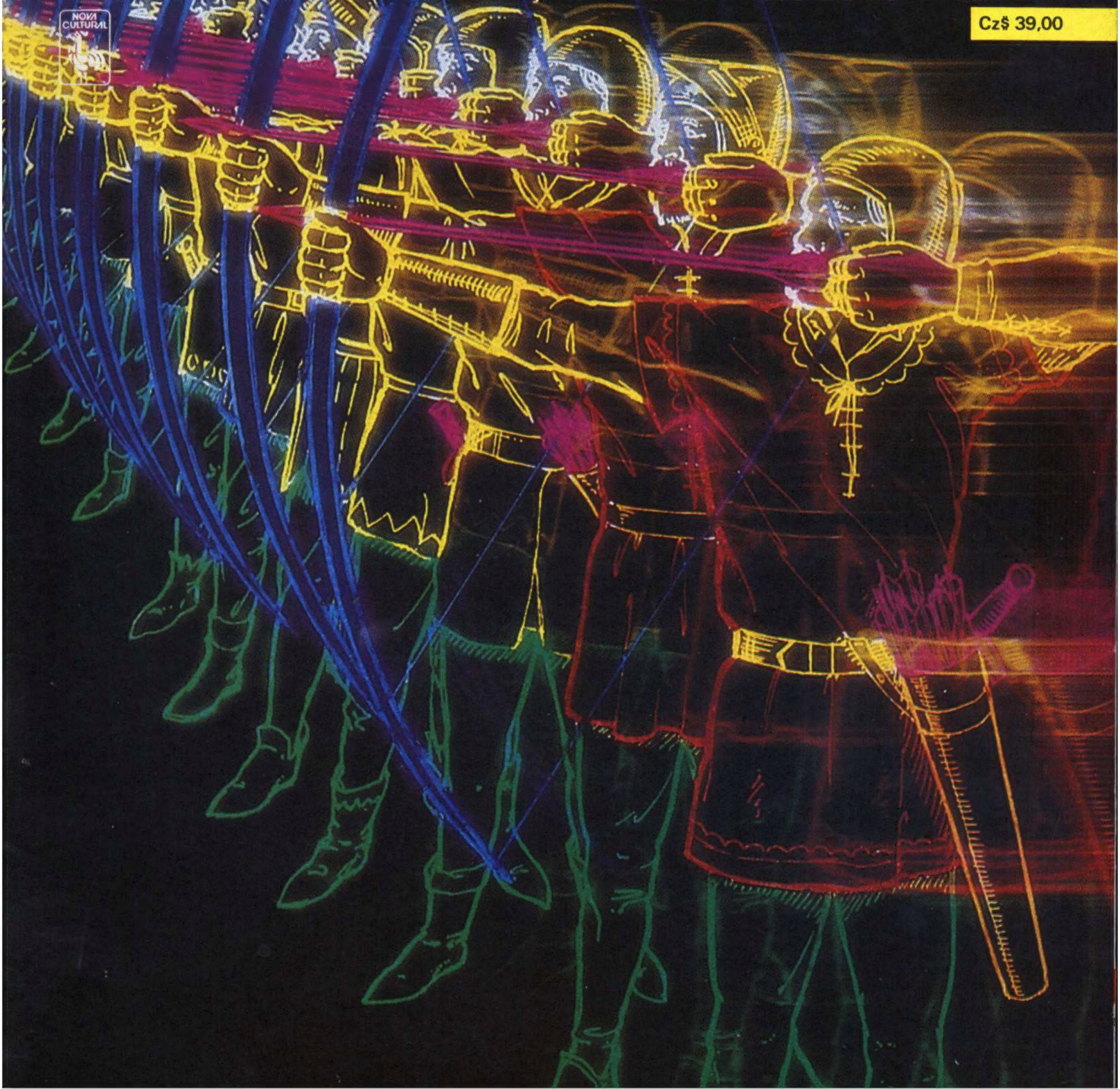
CURSO PRÁTICO 53 DE PROGRAMAÇÃO DE COMPUTADORES

# INPUT

PROGRAMAÇÃO BASIC – PROGRAMAÇÃO DE JOGOS – CÓDIGO DE MÁQUINA

NOVA CULTURAL

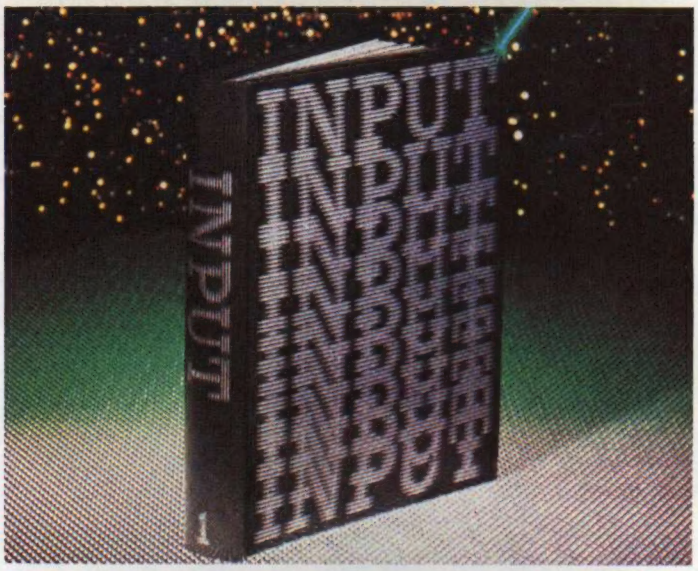
Cz$ 39,00



# INPUT

Vol. 4 N.º 53

## NESTE NÚMERO




### PROGRAMAÇÃO DE JOGOS

**JOGOS DE GUERRA: A ARTE DE COMANDAR**

Matriz da tropa: valores que afetam o comportamento das unidades. Acerto das variáveis. As quatro ordens: Fogo, Alto, Marche, Status. Rotina de comando. O efeito das ordens..... 1041

### APLICAÇÕES

**UM AMPLIADOR GRÁFICO**

Como desenhar. O ajuste da escala. Ampliação e redução. Como executar o programa. Detalhes escondidos. Um jogo divertido. Desenhos de precisão. Aplicações profissionais ..... 1049

### CÓDIGO DE MÁQUINA

**PERIPÉCIAS NO REINO DE NETUNO**

*Avalanche*: rotina do avanço da maré. Variável de atraso. Mar agitado. Impressão dos caracteres na tela. A rotina **CHARPR**..... 1056

## PLANO DA OBRA

"INPUT" é uma obra editada em fascículos semanais, e cada conjunto de 15 fascículos compõe um volume. A capa para encadernação de cada volume estará à venda oportunamente.

## COMPLETE SUA COLEÇÃO

Exemplares atrasados, até seis meses após o encerramento da coleção, poderão ser comprados, a preços atualizados, da seguinte forma: 1. PESSOALMENTE — Por meio de seu jornaleiro ou dirigindo-se ao distribuidor local, cujo endereço poderá ser facilmente conseguido junto a qualquer jornaleiro de sua cidade. Em **São Paulo**, os endereços são: rua Brigadeiro Tobias, 773, Centro; avenida Industrial, 117, Santo André; e no **Rio de Janeiro**: avenida Mem de Sá, 191/193, Centro. 2. POR CARTA — Poderão ser solicitados exemplares atrasados também por carta, que deve ser enviada para DINAP — Distribuidora Nacional de Publicações — Números Atrasados — Estrada Velha de Osasco, 132, Jardim Teresa — CEP 06000 — Osasco — SP. Não envie pagamento antecipado. O atendimento será feito pelo reembolso postal e o pagamento, incluindo as despesas postais, deverá ser efetuado ao se retirar a encomenda na agência do Correio. 3. POR TELEX — Utilize o nº (011) 33 670 DNAP.

Em **Portugal**, os pedidos devem ser feitos à Distribuidora Jardim de Publicações, Lda. — Qta. Pau Varais, Azinhaga de Fetais — 2 685, Camarate — Lisboa; Apartado 57 — Telex 43 069 JARLIS P.

**Atenção:** Após seis meses do encerramento da coleção, os pedidos serão atendidos dependendo da disponibilidade do estoque.

**Obs.:** Quando pedir livros, mencione sempre título e/ou autor da obra, além do número da edição.

## COLABORE CONOSCO

Encaminhe seus comentários, críticas, sugestões ou reclamações ao Serviço de Atendimento ao Leitor — Caixa Postal 9442, São Paulo — SP.

NOVA CULTURAL

**Editor**
VICTOR CIVITA

**REDAÇÃO**
**Diretor Editorial:** Carmo Chagas

**Editores Executivos:** Antonio José Filho, Berta Sztark Amar

**Editor Chefe:** Paulo de Almeida
**Editor de Texto:** Cláudio A. V. Cavalcanti
**Chefe de Arte:** Carlos Luiz Batista
**Assistentes de Arte:** Dagmar Bastos Sampaio, Grace Alonso Arruda, Monica Lenardon Corradi
**Secretária de Redação/ Coordenadora:** Stefania Crema
**Secretários de Redação:** Beatriz Hagström, José Benedito de Oliveira Damião, Maria de Lourdes Carvalho, Marisa Soares de Andrade, Mauro de Queiroz

**COLABORADORES**

**Consultor Editorial Responsável:** Dr. Renato M. E. Sabbatini (Diretor do Núcleo de Informática Biomédica da Universidade Estadual de Campinas)
**Execução Editorial:** DATAQUEST Assessoria em Informática Ltda., Campinas, SP
**Tradução, adaptação, programação e redação:** Abílio Pedro Neto, Aluísio J. Dornellas de Barros, Marcelo R. Pires Therezo, Marcos Huascar Velasco, Raul Neder Porrelli, Ricardo J. P. de Aquino Pereira
**Coordenação Geral:** Rejane Felizatti Sabbatini
**Editora de Texto:** Ana Lúcia B. de Lucena

**COMERCIAL**
**Diretor Comercial:** Roberto Martins Silveira
**Gerente Comercial:** Flávio Maculan
**Gerente de Circulação:** Denise Maria Mozol

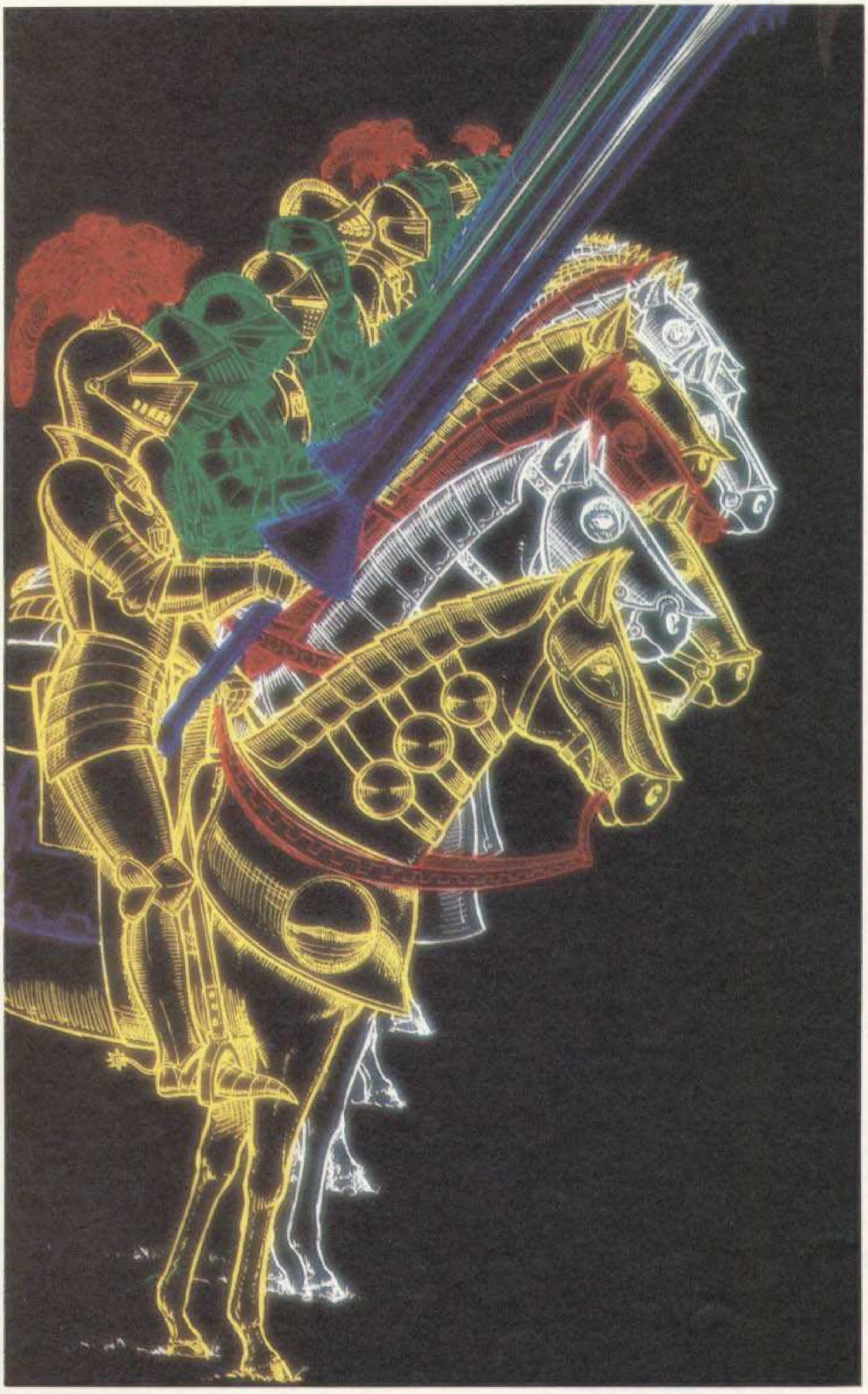
**PRODUÇÃO**
**Gerente de Produção:** João Stungis
**Coordenador de Impressão:** Atílio Roberto Bonon

**Preparador de Texto/Coordenador:** Eliel Silveira Cunha
**Preparadores de Texto:** Alzira Moreira Braz, Ana Maria Dilguerian, Levon Yacubian, Luciano Tasca, Maria Teresa Galluzzi, Maria Teresa Martins Lopes, Paulo Felipe Mendrone
**Revisor/Coordenador:** José Maria de Assis
**Revisoras:** Conceição Aparecida Gabriel, Isabel Leite de Camargo, Ligia Aparecida Ricetto, Maria de Fátima Cardoso, Nair Lucia de Britto
**Paste-up:** Anastase Potaris, Balduino F. Leite, Edson Donato

© Marshall Cavendish Limited 1984/85.
© Editora Nova Cultural Ltda., São Paulo, Brasil, 1986.
Edição organizada pela Editora Nova Cultural Ltda.
Av. Brigadeiro Faria Lima, nº 2000 - 3º andar
CEP 01452 - São Paulo - SP - Brasil
(Artigo 15 da Lei 5 988, de 14/12/1973).
Esta obra foi composta na AM Produções Gráficas Ltda. e impressa na Divisão Gráfica da Editora Abril S.A.

# JOGOS DE GUERRA: A ARTE DE COMANDAR

- O INÍCIO DA PARTIDA
- COMO DAR ORDENS À TROPA
- MOVIMENTO E DIREÇÃO
- STATUS
- O COMPUTADOR COMO JOGADOR



**Mobilize seu exército para a guerra, definindo uma estratégia de ação e disciplinando a tropa. Esses requisitos são fundamentais para o exercício da arte de comandar.**

Agora que as rotinas de posicionamento e movimentação das tropas já foram incorporadas ao programa, é preciso começar a dar ordens ao exército.

Diversos fatores tendem a influenciar o comportamento de cada unidade no campo de batalha. Os mais significativos dentre eles são os seguintes:

- Última ordem recebida pela unidade.
- Direção em que a unidade se move.
- Tipo de munição utilizado.
- Armaduras e demais equipamentos de que dispõe a unidade.
- Poder destrutivo inicial.
- Poder destrutivo no momento em que a batalha é travada.
- Moral da tropa.
- Posição estratégica assumida pela unidade.
- Terreno em que a batalha tem lugar.

Esses fatores correspondem aos nove elementos da matriz da tropa, que dimensionamos no artigo anterior. O terreno e a posição já foram definidos por ocasião da rotina de movimento. Neste artigo atribuiremos valores aos elementos restantes.

Ao começar o jogo, os valores iniciais adequados são colocados na matriz da tropa. Os tipos de armadura e de munição serão sempre os mesmos. Da mesma forma, o moral será também quase sempre o mesmo, embora possa sofrer algumas variações — os camponeses provavelmente nunca estarão muito dispostos a lutar (afinal, a guerra não é deles, mas dos barões feudais), enquanto os cavaleiros, para manter as aparências, nunca agirão covardemente. A capacidade destrutiva poderá ser muito diferente de uma partida para a outra. As ordens e as direções iniciais são determinadas pelo programa: todos começam parados ("ALTO"), de modo que não existe uma direção definida.

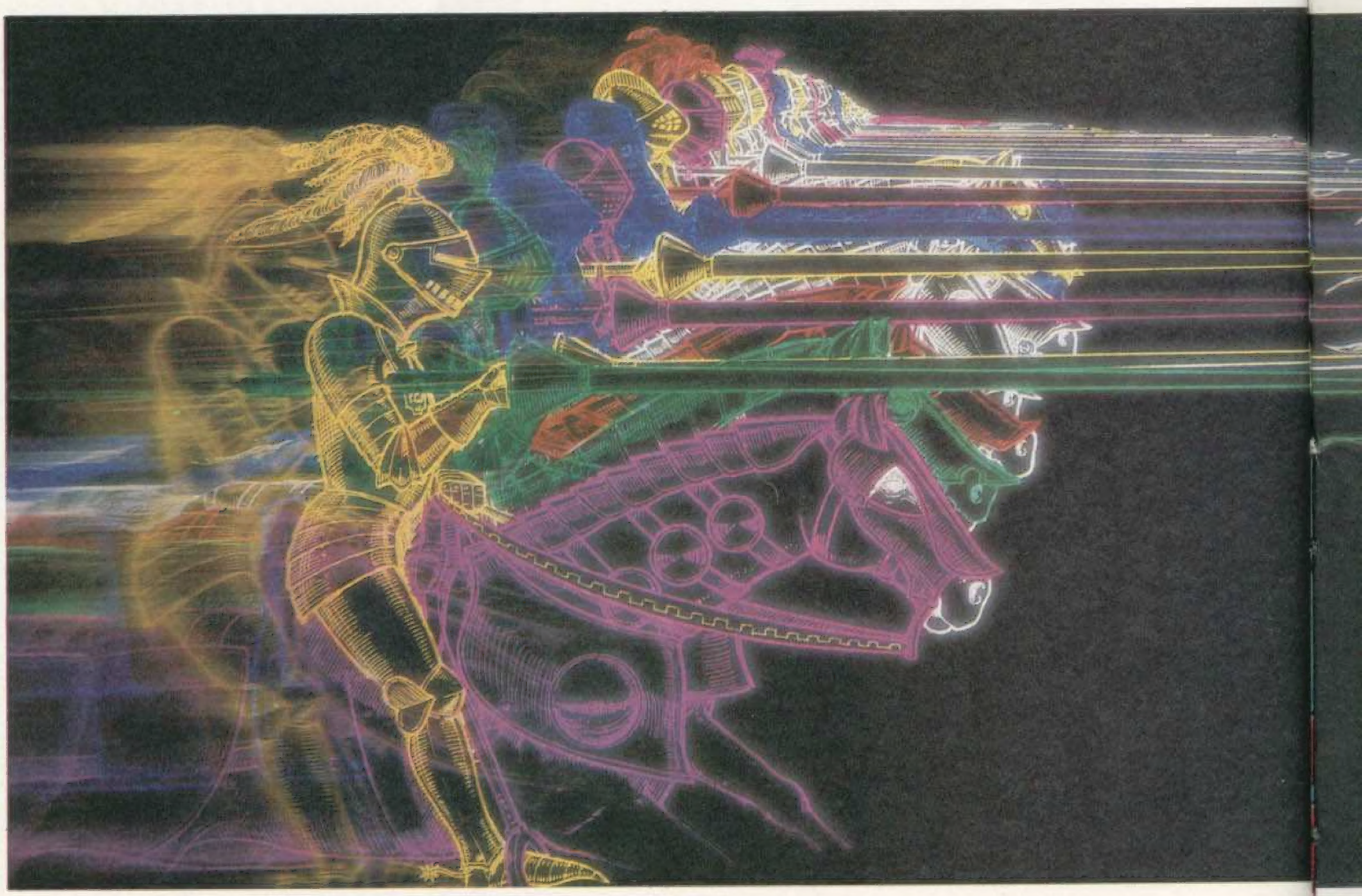
As rotinas que se seguem acertam o valor de diversas variáveis. Os dados são obtidos de linhas **DATA**, ou através de alguns cálculos, ou das duas maneiras. Quando o programa estiver completo e em funcionamento, poderemos modificar esses dados de acordo com nossas preferências.

## ABANDONE O QUARTEL-GENERAL

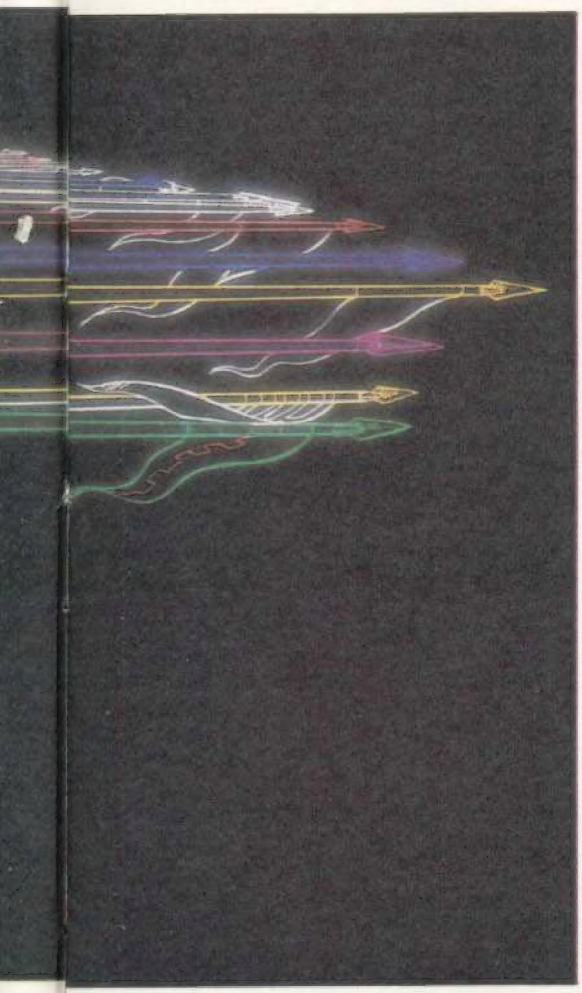
Estas rotinas acertam os valores iniciais dos elementos que ainda não foram definidos:

```
190 REM Inicializacao
200 LET vc=0: LET de=0
310 REM
320 PAPER 7
330 INK 0
340 CLS
360 DIM t$(8,12): DIM o$(5,12)
: DIM w$(5,9): DIM m$(5,12):
DIM a$(4,12): DIM r$(4,12):
DIM c(8)
380 FOR i=1 TO 5: READ o$(i),
w$(i),m$(i): NEXT i
390 FOR i=1 TO 8: READ t$(i):
NEXT i
400 FOR i=1 TO 4: READ a$(i),
r$(i): NEXT i
410 LET u$=CHR$ 148+CHR$ 149+
CHR$ 150+CHR$ 151+CHR$ 151+
CHR$ 152+CHR$ 152: LET u$=u$+
u$
415 LET x$="NnSs"
420 RETURN
2760 DATA "fogo","nada","covard
e","alto","arco","baixo"
2770 DATA "marche","espada","di
sposto","status","machado","bra
vo"
2780 DATA "retirada","lanca","v
alente"
2790 DATA "cavaleiros","sargent
os","lanceiros","arqueiros","ar
queiros","camponeses","campones
es"
2800 DATA "nada","campo","gibao
","vila","malha metalica","flor
esta","chapa metalica","montanh
a"
2810 DATA 5,4,3,5,3,3,4,3,2,3,3
,1,2,2,1,2,3,2,3,2,0,3,1,0
```

```
190 REM INICIO
200 VC=0:DE=0
340 CLS
360 DIM T$(8),O$(5),W$(5),M$(5)
,A$(4),R$(4)
380 FOR J=1 TO 5:READ O$(J),W$(
J),M$(J):NEXT J
390 FOR J=1 TO 8:READ T$(J):NEX
T J
400 FOR J=1 TO 4:READ A$(J),R$(
J):NEXT J
415 X$="NnSs"
420 RETURN
2760 DATA FOGO,NADA,COVARDE,ALT
O,ARCO,BAIXO
2770 DATA MARCHE,ESPADA,DISPOST
O,STATUS,MACHADO,BRAVO
2780 DATA RETIRADA,LANCA,VALENT
E
2790 DATA CAVALEIROS,SARGENTOS,
LANCEIROS,LANCEIROS,ARQUEIROS,A
RQUEIROS,CAMPONESES,CAMPONESES
2800 DATA NADA,CAMPO,GIBÃO,VILA
,MALHA METÁLICA,FLORESTA,PLACA
METÁLICA,MONTANHA
2810 DATA 5,4,3,5,3,3,4,3,2,3,3
,1,2,2,1,2,3,2,3,2,0,3,1,0
```



```
190  REM  INICIO
200 CL = 0:VC = 0:DE = 0
340  HGR2 : POKE  - 16301,0: GO
SUB 2540
360  DIM T$(8),O$(5),W$(5),M$(5
),A$(4),R$(4)
380  FOR I = 1 TO 5: READ O$(I)
,W$(I),M$(I): NEXT I
390  FOR I = 1 TO 8: READ T$(I)
: NEXT I
400  FOR I = 1 TO 4: READ A$(I)
,R$(I): NEXT I
410 U$ = "45667788"
415 X$ = "NNSS"
420  RETURN
2760  DATA  FOGO,NADA,COVARDE,A
LTO,ARCO,BAIXO,MARCHE,ESPADA,DI
SPOSTO
2780  DATA  STATUS,MACHADO,BRAV
O,RETIRADA,LANCA,VALENTE
2790  DATA  CAVALEIROS,SARGENTO
S,LANCEIROS,LANCEIROS,ARQUEIROS
,ARQUEIROS,CAMPONESES,CAMPONESE
S
2800  DATA  NADA,CAMPO,GIBAO,VI
LA,MALHA METALICA,FLORESTA,CHAP
A METALICA,MONTANHA
2810  DATA  5,4,3,5,3,3,4,3,2,
3,3,1,2,2,1,2,3,2,3,2,0,3,1,0
```



Os usuários do micro TK-2000 devem modificar as seguintes linhas do programa destinado ao Apple.

```
340  HGR2 : GOSUB 2540
```

```
310 REM
330 COLOR 4,2
340 PCLS
360 DIM T$(8),O$(5),W$(5),M$(5)
,A$(4),R$(4)
380 FOR J=1 TO 5:READ O$(J),W$(
J),M$(J):NEXT J
390 FOR J=1 TO 8:READ T$(J):NEX
T J
400 FOR J=1 TO 4:READ A$(J),R$(
J):NEXT
415 X$="NnSs"
420 RETURN
2760 DATA FOGO,NADA,COVARDE,ALT
O,ARCO,BAIXO
2770 DATA MARCHE,ESPADA,DISPOST
O,STATUS,MACHADO,BRAVO
2780 DATA RETIRADA,LANCA,VALENT
E
2790 DATA CAVALEIROS,SARGENTOS,
LANCEIROS,LANCEIROS,ARQUEIROS,A
RQUEIROS,CAMPONESES,CAMPONESES
2800 DATA NADA,CAMPO,GIBAO,VILA
,MALHA METALICA,FLORESTA,CHAPA
METALICA,MONTANHA
2810 DATA 5,4,3,5,3,3,4,3,2,3,3
,1,2,2,1,2,3,2,3,2,0,3,1,0
```

Os valores iniciais são obtidos com **READ** das linhas **DATA**, juntamente com as palavras equivalentes aos valores numéricos colocados na matriz. As palavras serão usadas na janela de texto para que a batalha se desenrole de maneira clara para o jogador.

## DÊ AS ORDENS

O jogo consiste basicamente em dar ordens às unidades — sem isto não haverá combate e, conseqüentemente, vitória ou derrota.

Existem quatro ordens básicas no jogo *Capa e Espada*: "Fogo", "Alto", "Marche" e "Status". A ordem de atirar se aplica somente aos arqueiros, que obedecerão mesmo que não haja unidades inimigas nas redondezas.

Dar ordens às tropas tem suas desvantagens: quanto mais real quisermos que seja um jogo de guerra, mais complicadas serão essas ordens. Inversamente, quanto mais simples for o jogo, mais fácil será o processo de comando. Assim, um jogo muito realista será mais difícil de jogar; e, se quisermos produzir um jogo fácil de jogar, teremos que sacrificar os detalhes. Depois de programar alguns jogos desse tipo, você será capaz de criar programas mais complexos.

## ROTINA DE COMANDO

A rotina de comando de *Capa e Espada* seleciona as unidades de cada jogador, uma a uma, para que estes possam dar as ordens. A unidade em questão tem sua cor modificada no mapa (em alguns computadores a mudança é sutil). Uma mensagem solicitando as novas ordens para essa unidade (indicada por seu número) é mostrada na janela de texto. Se o jogador não quiser alterar as ordens, basta pressionar a tecla N. Se S for pressionada, será mostrado um menu com as ordens possíveis.

Caso seja selecionada a ordem de atirar (**FOGO**) e a unidade não seja constituída de arqueiros, uma nova ordem será escolhida. O comando **ALTO** faz a unidade permanecer onde está e **MARCHE** exige a definição da direção, indicada pelas iniciais de norte, sul, leste e oeste. O programa não faz nenhum teste para verificar se a direção é válida; portanto, preste atenção ao mapa.

## COMO SELECIONAR UMA UNIDADE

Quando chega a vez do jogador, a primeira unidade é destacada por meio da mudança de cor; isso se repete com cada uma das outras sete unidades. O símbolo que muda de cor é a unidade que o jogador deve considerar: as ordens devem ou não ser modificadas? Quando a unidade é destacada, uma mensagem surge na janela de texto, mostrando a última ordem dada.

```
1380 REM unidade
1390 GOSUB 2540
1400 INK 0
1410 PRINT  FLASH 1;AT T(i,8),T
(i,9);u$(i)
1420 PRINT AT 17,0;"Unidade num
ero ";i;" ";t$(i);"      "
1430 PRINT AT 18,0;"Ultima orde
m:";o$(T(i,1));" ";
1440 IF T(i,1)=3 THEN  PRINT AT
 18,28;i$(T(i,2))
1450 REM Loop
1460 PRINT AT 19,0;"Deseja muda
```

```
r as ordens (S/N)?"
1465 LET y=0: LET y$=INKEY$
1466 IF y$="" THEN  GOTO 1465
1470 FOR k=1 TO 4
1475 IF x$(k)=y$ THEN  LET y=k
1480 NEXT k
1490 IF y=0 THEN  GOTO 1450
1500 RETURN
```

```
1380 REM UNIDADE
1390 GOSUB 2540
1410 LOCATE T(I,9),T(I,8):PRINT
 CHR$(U(I)+48);
1420 LOCATE 0,18:PRINT "UNIDADE
";I;":";T$(I);"       "
1430 LOCATE 0,19:PRINT "ULTIMA
ORDEM:";O$(T(I,1));
1440 IF T(I,1)=3 THEN PRINT I$(
T(I,2))
1460 LOCATE 0,20:PRINT "QUER MU
DAR AS ORDENS (S/N)?"
1465 YW=0:Y$=INKEY$
1466 IF Y$="" THEN 1465
1470 YW=INSTR(1,X$,Y$)
1490 IF YW=0 THEN 1460
1500 RETURN
```

```
1380  REM  UNIDADE
1390  GOSUB 2540
1410 X = T(I,9):Y = T(I,8):N =
 VAL ( MID$ (U$,I - (I > 8) * 8
,1)) + (I > 8) * 5: GOSUB 20000
1420  VTAB 21: PRINT "UNIDADE N
UMERO ";I;" ";T$(I)
1430  PRINT "ULTIMA ORDEM: ";O$
(T(I,1));" ";
1440  IF T(I,1) = 3 THEN  PRINT
  MID$ (I$,T(I,2),1): GOTO 1460
1450  PRINT
1460  PRINT "DESEJA MUDAR AS OR
DENS (S/N)?": GOSUB 30000
1465 YW = 0: GET Y$
1470  FOR K = 1 TO 4
1475  IF  MID$ (X$,K,1) = Y$ TH
EN YW = K
1480  NEXT K
1490  IF YW = 0 THEN 1450
1500  RETURN
20000 X = X * 2 - 2:N = N * 2:
FOR W = 0 TO 1:X = X + W:N = N
+ W: FOR V = 0 TO 7
20010  POKE T + (Y - 8 * (Y > 7
) - 8 * (Y > 15)) * 128 + 40 *
(Y > 7) + 40 * (Y > 15) + X + 1
024 * V, PEEK (EE + N * 8 + V)
- 128
20020  NEXT V,W: RETURN
```

Faça a seguinte modificação no programa do Apple:

```
1460  PRINT "DESEJA MUDAR AS OR
DENS (S/N)?"
```

```
1380 REM UNIDADE
1390 GOSUB 2540
1400 COLOR 4
1410 DRAW"BM"+STR$(T(I,9)*8)+",
"+STR$(T(I,8)*8):UU=VAL(MID$(U$
,I,1)):A$=UC$(UU):GOSUB 3030
1420 DRAW"BM0,144":A$="UNIDADE"
+SR$(I)+" "+T$(I)+"        ":GOS
UB 3190
1430 DRAW"BM0,152":A$="ORDENS S
AO PARA   "+O$(T(I,1))+" ":GOSU
B 3190
1440 IF T(I,1)=3 THEN DRAW "BM1
60,152":A$=MID$(I$,T(I,2),1):GO
SUB 3190
1450 REM
1460 DRAW"BM0,160":A$="DESEJA M
UDAR AS ORDENS (S/N)":GOSUB 319
0
1465 Y=0:Y$=INKEY$
1466 IF Y$="" THEN 1465
1470 Y=INSTR(1,X$,Y$)
1490 IF Y=0 THEN 1450
1500 RETURN
3000 X9=PEEK(200):Y9=PEEK(202):
LINE(X9,Y9)-(X9+7,Y9+7),PRESET,
BF
3010 POKE 200,X9:POKE 202,Y9:DR
AW A$
3020 RETURN
3030 X9=PEEK(200):Y9=PEEK(202):
C9=PEEK(178):COLOR 2:LINE(X9,Y9
)-(X9+7,Y9+7),PSET,BF
3040 POKE 178,C9:GOTO 3010
3050 REM DADOS PARA LERAS E DIG
ITOS
3060 DATA BR4,BDD5RU3NR2U2ERFND
5BR4BU,D6R3EUHEUHBR5,NR5D6R5BR3
BU6,NR3D6R3EU4BUBR4,NR5D3NR3D3R
5BR3BU6,NR5D3NR3D3BR8BU6,NR5D6R
5U2BU4BR3
3070 DATA ND6D3R5D3U6BR3,R5L2D6
L3R5BR3BU6,BD5RFREU5L2BR6,ND6D3
R3FD2BU4U2BR4,D6R5BR3BU6,ND6DR5
NL5UBR3,ND6DR2D3R3D2U6BR3
3080 DATA D6R5U6NL5BR3,D6U3R5U3
NL5BR3,D6UR3FRBU2U4NL5BR3,D6U2R
3FDUBU2NL2U3NL4BR4,NR5D3R5D3NL5
BU6BR3,R5L2ND6BR5,D6R5U6BR3
3090 DATA D4RFDRUEU4BR4,D6UR5DU
6BR3,D2RFGND2ERFND2HEU2BR4,D2RF
D3RU3EU2BR4,R4D2GLGD2R4BU6BR3
3100 DATA BR3LGD4FREU4BR4BU,BR2
DNL2D5L2R5BU6BR3,BDRERFDGL2D3R4
BU6BR3,BDRERFDGFDGLHLBU5BR8,D4R
5UD3BU6BR3,NR5D2R3FD2GLHLBU5BR8
,BDBR5LHLGD2NR2D2FREUBU4BR4
3110 DATA R5D2LGLGD2BR7BU6,BR3L
GDFGDFREUHEUBUBR4,BR3LGDFR2D2GL
HLBU2BR4U2BUBR4
3120 REM MATRIZ DE CARACTERES
3130 DIM LE$(26)
3140 FOR K9=0 TO 26:READ LE$(K9
):NEXT
3150 FOR K9=0 TO 9:READ NU$(K9)
:NEXT
3170 RETURN
3180 REM ROTINA PARA IMPRIMIR A
$
3190 FOR K9=1 TO LEN(A$)
3200 B$=MID$(A$,K9,1)
3210 IF B$>="0" AND B$<="9" THE
N DRAW NU$(VAL(B$)):GOTO 3240
3220 IF B$=" " THEN N9=0 ELSE N
9=ASC(B$)-64
3230 DRAW LE$(N9)
3240 NEXT
3250 RETURN
```

As ordens para a unidade destacada são mostradas na janela de texto. O jogador é indagado, então, sobre se deseja mudar as ordens.

O micro TRS-Color possui linhas adicionais (3000 a 3250) destinadas a desenhar as letras e os números na tela de alta resolução.

## A ARTE DE COMANDAR

Esta rotina mostra um menu com as ordens possíveis:

```
1900 REM  acao
1910 GOSUB 2540
1920 PRINT AT 18,0;"Opcoes :"
1930 FOR j=1 TO 4
1940 PRINT AT 17+j,8;o$(j,1);"-
";o$(j)
1950 NEXT j
1960 REM loop
1962 LET a=0
1965 LET f$="FfAaMmSs"
1970 LET g$=INKEY$: IF g$="" TH
EN  GOTO 1970
1975 FOR k=1 TO 8
1980 IF f$(k)=g$ THEN  LET a=IN
T ((k+1)/2)
1985 NEXT k
1990 IF a<=0 THEN  GOTO 1960
2000 IF i<>6 AND i<>5 AND a=1 T
HEN  GOSUB 2540: PRINT AT 18,8;
"Nao ha arcos": GOSUB 2410: GOT
O 1910
2010 IF a=4 THEN  GOSUB 2440: R
ETURN
2020 LET T(i,1)=a
2030 IF a=3 THEN  GOSUB 2050
2040 RETURN
```

```
1900 REM AÇÃO
1910 GOSUB 2540
1920 LOCATE 0,19:PRINT "OPÇÕES:
";
1930 FOR J=1 TO 4
1940 LOCATE 8,18+J:PRINT LEFT$(
O$,2);"-";O$(J)
1950 NEXT J
1960 A=0
1965 F$="FHMS"
1970 G$=INKEY$:IF G$="" THEN 19
70
1980 A=INSTR(1,F$,G$)
```

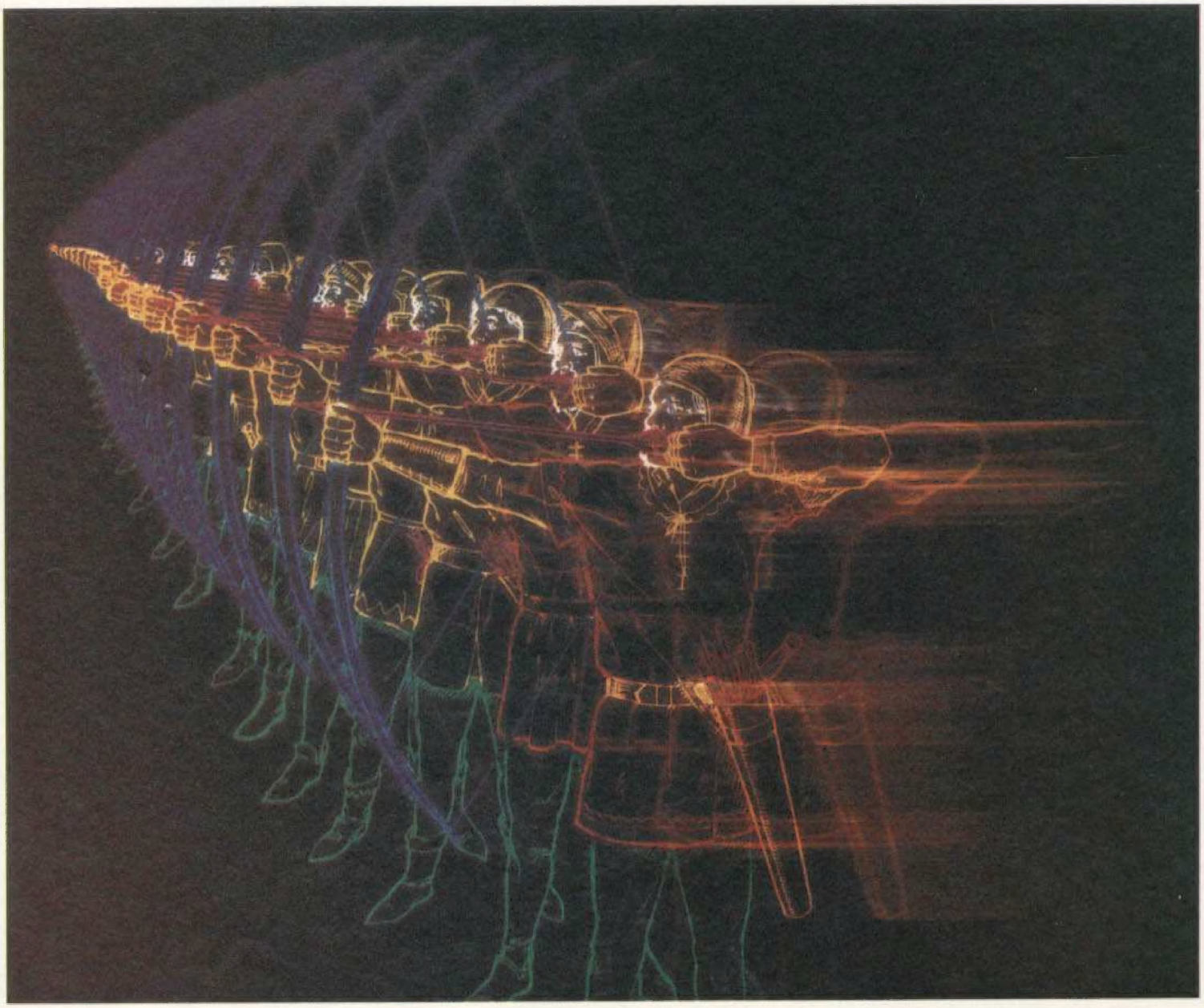
```
1990 IF A<=0 THEN 1960
2000 IF I<>6 AND I<>5 AND A=1 T
HEN GOSUB 2540:LOCATE 8,19:PRIN
T "SEM ARCOS":GOSUB 2410:GOTO 1
910
2010 IF A=4 THEN GOSUB 2440:RET
URN
2020 T(I,1)=A
2030 IF A=3 THEN GOSUB 2050
2040 RETURN
```

```
1900  REM  ACAO
1910  GOSUB 2540
1920  VTAB 21: PRINT "OPCOES: "
;
1930  FOR J = 1 TO 4
1940  HTAB 10: PRINT  LEFT$ (O$
(J),1);" - ";O$(J);: IF J < 4 T
HEN  PRINT " "
1950  NEXT J: GOSUB 30000
1960 F$ = "FFAAMMSS"
1970  GET G$
1973 A = 0
1975  FOR K = 1 TO 8
1980  IF  MID$ (F$,K,1) = G$ TH
EN A =  INT ((K + 1) / 2)
1985  NEXT K
1990  IF A <  = 0 THEN 1970
2000  IF I <  > 6 AND I <  > 5
AND A = 1 THEN  GOSUB 2540: PRI
NT "NAO HA ARCOS": GOSUB 30000:
 GOSUB 2410: GOTO 1910
2010  IF A = 4 THEN  GOSUB 2440
: RETURN
2020 T(I,1) = A
2030  IF A = 3 THEN  GOSUB 2050
2040  RETURN
```

Modificações para o TK-2000:

```
1950  NEXT J
2000  IF I <  > 6 AND I <  > 5
AND A = 1 THEN  GOSUB 2540: PRI
NT "NAO HA ARCOS": GOSUB 2410:
GOTO 1910
```

```
1900 REM SELECIONA ACAO
1910 GOSUB 2540
1920 DRAW"BM0,152":A$="OPCOES
   ":GOSUB 3190
1930 FOR J=1 TO 4
1940 DRAW"BM104,"+STR$(144+J*8)
```



```
:A$=LEFT$(O$(J),1)+"  "+O$(J):G
OSUB 3190
1950 NEXT J
1960 REM
1962 A=0
1965 F$="FAMS"
1970 G$=INKEY$:IG G$="" THEN 19
70
1980 A=INSTR(1,F$,G$)
1990 IF A<=0 THEN 1960
2000 IF I<>6 AND I<>5 AND A=1 T
HEN GOSUB 2540:DRAW"BM64,152":A
$="NAO HA ARCOS":GOSUB 3190:GOS
UB 2410:GOTO 1910
2010 IF A=4 THEN GOSUB 2440:RET
URN
2020 T(I,1)=A
2030 IF A=3 THEN GOSUB 2050
2040 RETURN
```

As linhas 1920 a 1950 mostram as opções na tela. As linhas 1965 a 1985 obtêm a resposta do teclado e colocam o número equivalente à ordem selecionada na variável **A**: 0 é "fogo", 1 é "alto", 2 é "marche" e 3 é "status".

A ordem de atirar só será possível por meio de uma rotina do próximo artigo da série. As opções de marchar e de status serão dadas pelas próximas rotinas. A ordem "ALTO" não precisa de uma rotina especial, já que nada acontece com a unidade. O elemento da matriz da tropa que indica a direção em que se movimentava a unidade não deve ser modificado porque é utilizado na rotina de combate.

A linha 2020 coloca o valor da variável **A** na matriz da tropa; o número da ordem corresponde ao primeiro elemento da matriz.

## UMA NOVA DIREÇÃO

Se o jogador ordenar a uma unidade que se movimente, o programa solicitará uma direção. Esta rotina cuida das opções possíveis:

```
2050 REM  Direcao
2055 GOSUB 2540
2060 PRINT AT 17,0;"Para onde (
NSLO) ?"
2065 LET g=0
2070 REM loop
2080 LET g$=INKEY$: IF g$="" TH
EN  GOTO 2080
2090 IF CODE (g$)>90 THEN  LET
g$=CHR$ (CODE (g$)-32)
2095 FOR k=1 TO 4
2100 IF i$(k)=g$ THEN  LET g=k
2105 NEXT k
2110 IF g=0 THEN  GOTO 2070
2120 LET T(i,2)=g
2130 RETURN
```

```
2050 REM DIREÇAO
2055 GOSUB 2540
2060 LOCATE 0,18:PRINT "PARA ON
DE (NSLO)?"
2065 G=0
2070 REM
2075 REM
2080 G$=INKEY$:IF G$="" THEN 20
80
2100 G=INSTR(1,I$,G$)
2110 IF G=0 THEN 2070
2120 T(I,2)=G
2130 RETURN
```

```
2050  REM  DIRECAO
2055  GOSUB 2540
2060  VTAB 21: PRINT "PARA ONDE
 (NSLO)?"
2065  GOSUB 30000
2070 G = 0
2080  GET G$
2090  IF  ASC (G$) > 90 THEN G$
 =  CHR$ ( ASC (G$) - 32)
2095  FOR K = 1 TO 4
2100  IF  MID$ (I$,K,1) = G$ TH
EN G = K
2105  NEXT K
2110  IF G = 0 THEN 2070
2120 T(I,2) = G
2130  RETURN
```

Modificações a fazer no programa:

```
2065  REM
```

```
2050 REM DIRECAO
2055 GOSUB 2540
2060 DRAW"BM0,144":A$="QUAL A D
IRECAO       N S L O ":GOSUB 31
90
2065 G=0
2070 REM
2080 G$=INKEY$:IF G$="" THEN 20
80
2100 G=INSTR(1,I$,G$)
2110 IF G<=0 THEN 2070
2120 T(I,2)=G
2130 RETURN
```

A rotina verifica se a letra teclada pelo jogador é N, S, L ou O. Se for uma delas, a linha 2120 colocará o número do movimento equivalente no segundo elemento da matriz da tropa.

## STATUS

Para conferir a situação de uma unidade, o jogador terá esta rotina.

```
2440 REM status
2450 GOSUB 2540
2470 PRINT AT 18,0;"Arma: ";w$(
T(i,3))
2480 PRINT AT 18,15;"Armadura:
";a$(T(i,4))
2490 PRINT AT 19,0;"Poder: ";T(
i,7)
2500 PRINT AT 19,14;"Moral: ";m
$(T(i,5))
2510 PRINT AT 20,0;"Terreno: ";
r$(m(T(i,8),T(i,9))+1)
2520 GOSUB 2410
2530 RETURN
2560 PRINT AT 17,0;"UNIDADE ";i
;"    Tipo: ";t$(i)
```

```
2440 REM STATUS
```

```
2450 GOSUB 2540
2460 LOCATE 0,18:PRINT "UNIDADE
";I,"TIPO:";T$(I)
2470 LOCATE 0,19:PRINT "ARMA:";
W$(T(I,3))
2480 LOCATE 15,19:PRINT "ARMADU
RA:";A$(T(I,4))
2490 LOCATE 0,20:PRINT "PODER:"
;T(I,7)
2500 LOCATE 15,20:PRINT "MORAL:
";M$(T(I,5))
2510 LOCATE 0,20:PRINT "POSIÇÃO
:";R$(M(T(I,8),T(I,9))+1)
2520 GOSUB 2410
2530 RETURN
```

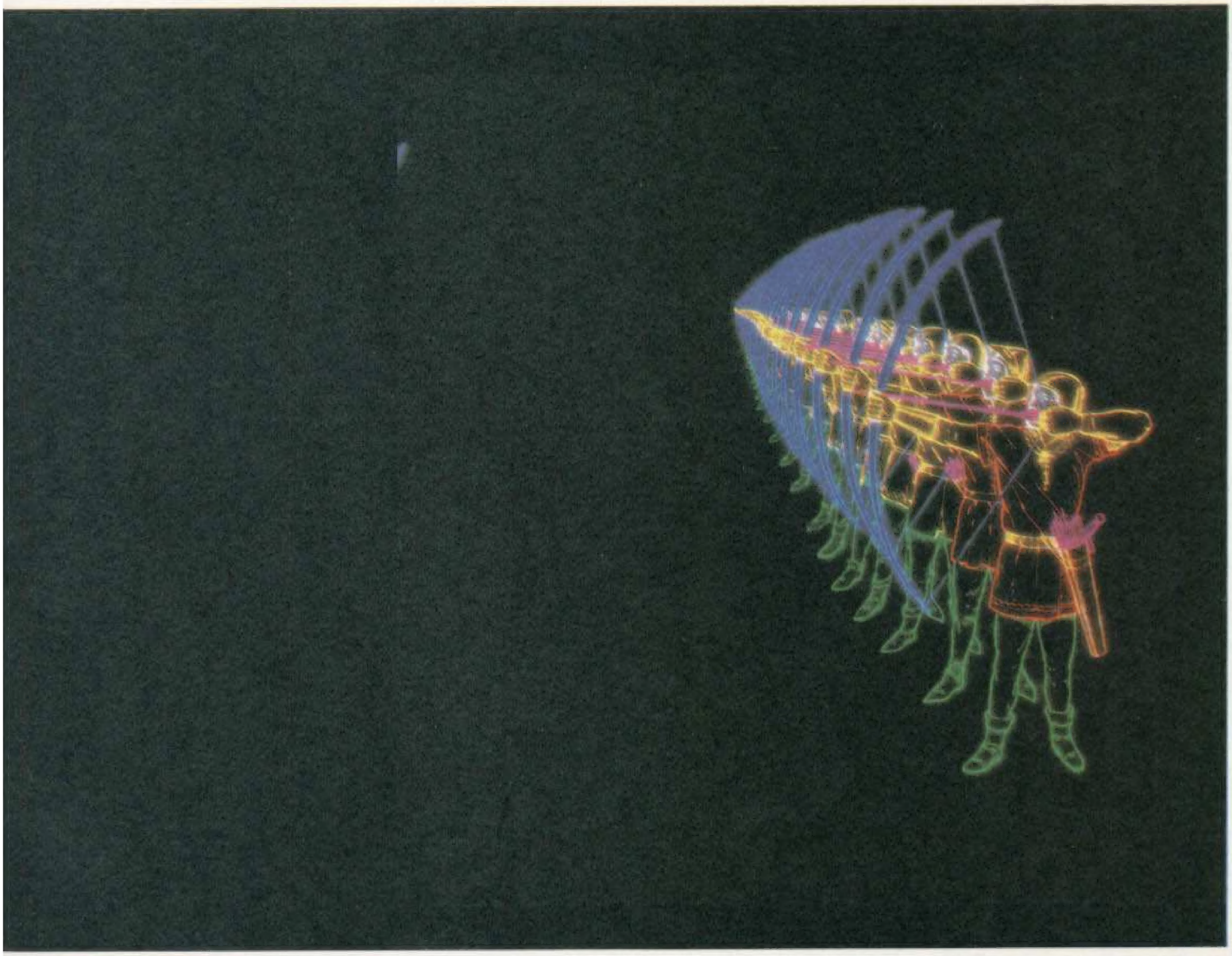
```
2440  REM  STATUS
2450  GOSUB 2540
2460  VTAB 21: PRINT "UNIDADE "
;I;".";"TIPO: "T$(I)
2470  PRINT "ARMA: ";W$(T(I,3))
;".";
2480  PRINT "ARMADURA: ";A$(T(I
,4));"."
2490  PRINT "PODER: ";T(I,7);".
";
2500  PRINT "MORAL: ";M$(T(I,5)
);"."
2510  PRINT "TERRENO: ";R$(M(T(
I,8),T(I,9)) + 1)
2520  GOSUB 2410
2530  RETURN
```

Modificações para o TK-2000:

```
2545  HCOLOR= 0: FOR ER = 160 T
O 191
2546  HPLOT 0,ER TO 279,ER: NEX
T
```

```
2440 REM  STATUS
2450 GOSUB 2540
2460 DRAW"BM0,144":A$="UNIDADE"
+STR$(I)+"    TIPO "+T$(I):GOSU
B 3190
2470 DRAW"BM0,152":A$="ARMA "+W
$(T(I,3)):GOSUB 3190
2480 DRAW"BM120,152":A$="ARMADU
RA "+A$(T(I,4)):GOSUB 3190
2490 DRAW"BM0,160":A$="PODER "+
SR$(T(I,7)):GOSUB 3190
2500 DRAW"BM120,160":A$="MORAL
"+M$(T(I,5)):GOSUB 3190
2510 DRAW"BM0,168":A$="TERRENO
"+R$(M(T(I,8),T(I,9))+1):GOSUB
3190
2520 GOSUB 2410
2530 RETURN
```

Todos os elementos pertencentes à matriz da tropa (ou as palavras correspondentes) que se referem à unidade que o jogador está querendo conferir são mostrados na tela.



## O EFEITO DAS ORDENS

Esta rotina informa ao jogador quando cada unidade obedece às ordens.

```
1020 REM  cumpre
1030 FOR i=1 TO 16
1032 IF t(i,1)>3 THEN  GOTO 114
0
1035 INK 0: GOSUB 2540: PRINT A
T 17,0;"Unidade ";i;" entra em
acao !"
1040 LET cl=1: IF i>8 THEN  LET
 cl=2: INK cl
1050 IF T(i,1)=3 THEN  LET b=1:
 GOSUB 1160
1055 IF T(i,1)=2 THEN  GOTO 114
0
1060 IF T(i,1)=1 THEN  LET sh=i
: GOSUB 1710: GOTO 1140
1070 FOR f=-1 TO 1
1080 FOR g=-1 TO 1
1090 FOR e=1 TO 16
1100 IF (T(i,8)+f=T(e,8)) AND (
T(i,9)+g=T(e,9)) AND T(e,1)<>5
THEN  LET us=i: LET th=e: GOSUB
 1510
1110 NEXT e
1120 NEXT g
1130 NEXT f
1140 NEXT i
1150 RETURN
```

```
1020 REM CUMPRE
1030 FOR I=1 TO 16
1032 IF T(I,1)>3 THEN 1140
1035 GOSUB 2540:LOCATE 0,18:PRI
NT "UNIDADE";I;"OBEDECE"
1040 IF T(I,1)>3 THEN 1140
1050 IF T(I,1)=3 THEN B=I:GOSUB
 1160
1055 IF T(I,1)=2 THEN 1140
1060 IF T(I,1)=1 THEN SH=I:GOSU
B 1710:GOTO 1140
1070 FOR F=-1 TO 1
1080 FOR G=-1 TO 1
1090 FOR E=1 TO 16
1100 IF (T(I,8)+F=T(E,8)) AND(T
(I,9)+G=T(E,9)) AND T(E,1)<>5 T
HEN US=I:TH=E:GOSUB 1510
1110 NEXT E
1120 NEXT G
1130 NEXT F
1140 NEXT I
1150 RETURN
```

```
1020  REM  CUMPRE
1030  FOR I = 1 TO 16
1032  IF T(I,1) > 3 THEN 1140
1035  GOSUB 2540: VTAB 21: PRIN
T "A UNIDADE ";I;" OBEDECE": GO
SUB 30000
1040  LET CL = 0: IF I > 8 THEN
 CL = 5
1050  IF T(I,1) = 3 THEN B = I:
 GOSUB 1160
1055  IF T(I,1) = 2 THEN 1140
1060  IF T(I,1) = 1 THEN SH = I
: GOSUB 1710: GOTO 1140
1070  FOR F =  - 1 TO 1
1080  FOR G =  - 1 TO 1
1090  FOR E = 1 TO 16
1100  IF (T(I,8) + F = T(E,8))
AND (T(I,9) + G = T(E,9)) AND T
(E,1) <  > 5 THEN US = I:TH = E
: GOSUB 1510
1110  NEXT E
1120  NEXT G
1130  NEXT F
1140  NEXT I
1150  RETURN
```

Modificações necessárias:

```
1035  GOSUB 2540: VTAB 21: PRIN
T "A UNIDADE ";I;" OBEDECE"
```

```
1020 REM CUMPRE
1030 FOR I=1 TO 16
1032 IF T(I,1)>3 THEN 1140
1035 COLOR 4:GOSUB 2540:DRAW "B
M0,144":A$="UNIDADE"+STR$(I)+"E
NTRA EM ACAO !":GOSUB 3190
1040 CL=3:IF I>8 THEN CL=4:COLO
R CL
1050 IF T(I,1)=3 THEN B=1:GOSUB
 1160
1055 IF T(I,1)=2 THEN 1140
1060 IF T(I,1)=1 THEN SH=I:GOSU
B 1710:GOTO 1140
1070 FOR F=-1 TO 1
1080 FOR G=-1 TO 1
1090 FOR E=1 TO 16
1100 IF (T(I,8)+F=T(E,8)) AND (
T(I,9)+G=T(E,9)) AND T(E,1)<>5
THEN US=1:TH=E:GOSUB 1510
1110 NEXT E,G,F
1140 NEXT I
1150 RETURN
```

O funcionamento dessa rotina é particularmente simples. Ela toma cada uma das unidades e executa as ordens. Só haverá combate se uma das unidades se mover (a que se moveu por último é considerada atacante). Quando uma ordem de "alto" é detectada, tudo permanece como está, passando-se à próxima opção. Quando ordens de "fogo" são encontradas, a rotina de tiro é chamada. Se as ordens forem de "marche", será chamada a rotina **MOVE**.

Após a realização de um movimento, a rotina verifica as posições adjacentes à nova localização da unidade em busca de tropas inimigas (linhas 1070 a 1150). Se alguma delas for encontrada, será chamada a rotina de combate, que apresentaremos no próximo artigo.

## A VEZ DO COMPUTADOR

Por enquanto, o computador dará ordens ao acaso. Na última parte desta série, veremos como transformá-lo em um adversário inteligente. A rotina seguinte limita-se a tornar o programa possível de ser jogado.

```
2140 REM inimigo
2150 LET T(e,2)=3
2160 LET T(e,1)=FN r(3)
2170 IF T(e,1)=1 AND T(e,3)<>2
THEN  GOTO 2160
2180 IF T(e,1)=3 THEN  IF FN r(
2)=1 THEN  LET T(e,2)=FN r(4)
2190 RETURN
```

```
2140 REM INIMIGO
2150 T(E,2)=3
2160 T(E,1)=FN R(3)
2170 IF T(E,1)=1 AND T(E,3)<>2
THEN 2160
2180 IF T(E,2)=3 THEN IF FN R(2
)=1 THEN T(E,2)=FN R(4)
2190 RETURN
```

```
2140  REM  INIMIGO
2150 T(E,2) = 3
2160 T(E,1) =  FN R(3)
2170  IF T(E,1) = 1 AND T(E,3)
<  > 2 THEN 2160
2180  IF T(E,1) = 3 THEN  IF  F
N R(2) = 1 THEN T(E,2) =  FN R(
4)
2190  RETURN
```

```
2140 REM  INIMIGO
2150 T(E,2)=3
2160 T(E,1)=RND(3)
2170 IF T(E,1)=1 AND T(E,3)<>2
THEN 2160
2180 IF T(E,1)=3 AND RND(2)=1 T
HEN T(E,2)=RND(4)
2190 RETURN
```

A rotina gera um número aleatório na linha 2160 que decide o tipo de ação das unidades do computador na jogada. Se o computador decidir mover alguma unidade, haverá uma tendência ao movimento em direção sul (linha 2150).

# UM AMPLIADOR GRÁFICO

- COMO DESENHAR
- AJUSTE A ESCALA
- COMO AMPLIAR E REDUZIR
- DETALHES ESCONDIDOS
- DESENHOS DE PRECISÃO

**O programa deste artigo pode reduzir um desenho a proporções diminutas ou expandi-lo em escala astronômica. Utilize-o como jogo ou para realizar desenhos muito detalhados.**

O programa que acompanha este artigo é muito mais divertido do que os programas aplicativos habituais. De fato, ele pode ser usado, seja como jogo, seja como aplicação.

Ele permite desenhar e, em seguida, ampliar ou reduzir, até um nível microscópico, partes de uma figura para obter maiores detalhes. É possível, além disso, mudar a escala a qualquer momento, usando ampliações de milhares de vezes ou até mesmo de centenas de milhares. Assim, uma enorme quantidade de detalhes que seriam imperceptíveis a olho nu numa figura de proporções reduzidas aparece de modo bem claro quando ela é submetida a uma ampliação por meio de um *zoom*.

Imagine-se com um poderoso microscópio, penetrando cada vez mais em direção à estrutura da imagem. Os detalhes desta aparecerão à medida que a ampliação for aumentando e, inversamente, desaparecerão quando ela sofrer uma redução de certa magnitude.

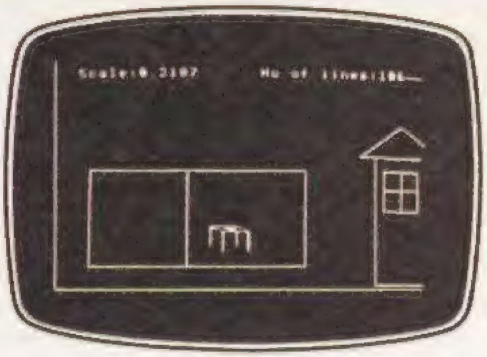
Para tornar ainda mais claro esse processo, podemos considerar o seguinte exemplo: se começarmos a fotografar nossa casa e nos afastarmos pouco a pouco até uma distância de centenas de milhares de quilômetros, os objetos — primeiro a casa, depois a rua, o bairro, a cidade, o país e mesmo a Terra — desaparecerão gradualmente das imagens retidas pela câmara. Tais efeitos gráficos podem ser conseguidos com o programa deste artigo.

Uma maneira de usar o programa é transformá-lo em um jogo para duas pessoas. Uma delas faz um desenho que esconde, em algum lugar discreto, um "tesouro", de tamanho diminuto. A outra deve procurar o tesouro. Essa tarefa pode se revelar surpreendentemente difícil. Imaginemos que o jogador comece sua busca em um quadrado de 10 cm de lado. Se este for ampliado 5 000 vezes, o quadrado original se transformará em uma área enorme de cerca de 250 000 $m^2$ — espaço suficiente para esconder qualquer coisa.

O programa se presta ainda a aplicações em diferentes áreas profissionais. Desenhos minuciosos, por exemplo, podem ser produzidos traçando-se os detalhes em uma escala de grandes proporções e depois reduzindo-se a figura para um tamanho normal. Desenhos técnicos muito precisos podem ser obtidos trabalhando-se em um setor de cada vez. O programa também é útil como elemento de apoio no campo pedagógico. Tomemos como exemplo uma lição geográfica. A posição das maiores cidades pode ser marcada no mapa em uma escala muito reduzida. No tamanho normal, ficariam só como pontos residuais e apareceriam em detalhe apenas quando se usasse um *zoom* no local adequado.

O programa oferece também a possibilidade de se escrever nomes. Embora não permita o uso de letras normais, é possível desenhá-las. Para isso, recorra a uma escala bem grande e depois reduza as letras para o tamanho adequado.

Ampliando este desenho...



você poderá olhar através da vidraça...

e verificar que há um livro sobre a mesa!

## COMO EXECUTAR O PROGRAMA

Ao ser executado, o programa apresentará a tela em branco e um pequeno cursor no centro dela. Você encontrará também dois números: o da esquerda indica a escala da tela atual — 1 para começar; o da direita mostra o número de linhas desenhadas. Esse número é limitado a um máximo de seiscentos no início do programa, que é controlado pelo teclado e muito fácil de usar.

No MSX, no Spectrum e no TRS-Color, o cursor é controlado pelas setas do teclado. Nos outros micros, as teclas são: I, para cima; M, para baixo; J, para a esquerda e K, para a direita. Pressionando-se simultaneamente essas teclas — **<SYMBOL-SHIFT>** no Spectrum, ou **<CLEAR>** no TRS-Color — acelera-se o cursor.

As outras teclas usadas são L, M, C, D, E, S e Z. Veja o que elas fazem: L desenha uma linha junto ao cursor; M move o cursor sem desenhar (nos micros da linha Apple, use ; em vez de M para mover o cursor). Pressionar C leva o cursor ao centro da tela; D cancela linhas. O número de linhas a ser apagado será perguntado pelo computador. Pressione S para ter acesso a um arquivo de saída que permita a armazenagem de seu desenho em disco ou fita.

Agora, a parte mais interessante. A tecla Z (para zoom) permite a alteração da escala. É digitada sempre com um número: 2, por exemplo, dobra a escala e 0.5 a reduz ao meio. O desenho é refeito na nova escala e *centrado no cursor*. Assim, você deve deixar o cursor sempre no meio da área que deseja ampliar.

Usar 0 como escala não altera o desenho, enquanto 1 redesenha a figura centrada no cursor, reproduzindo-a na escala anterior (no Apple, 0 centra o desenho e 1 retorna a imagem para a escala 1). Um número negativo produz uma imagem em espelho do desenho. As mudanças de escala são cumulativas. Assim, um aumento de 2 seguido de outro de 2 leva a ampliação para 4.

Note que uma escala muito pequena no TRS-Color, como 0.0001, é convertida para 10E-4, mas aparece na tela como 104. Esta é uma limitação da rotina que desenha números na tela e não afeta a maneira com que o desenho é feito.

```
 10 DEF FN a(x)=(x/256)+128
 20 DEF FN b(x)=(x/256)+85
 30 BORDER 0: INK 7: PAPER 0:
CLS
 50 LET ln=1: LET sc=1: LET l=
600
 60 LET f=0: LET x=0: LET y=0:
LET zoom=1
 70 DIM a(l+1): DIM b(l): DIM
c(l): DIM d(l)
 80 LET a(l)=x: LET b(l)=y
 90 LET c(l)=x: LET d(l)=y
100 LET i$=INKEY$
110 IF i$="l" AND f AND l>ln
THEN  SOUND 0.1,0: PLOT FN a(a
(ln)),FN b(b(ln)): DRAW FN a(c
(ln))-PEEK 23677,FN b(d(ln))-
PEEK 23678: LET ln=ln+1: LET a
(ln)=x: LET b(ln)=y: LET c(ln)
=x: LET d(ln)=y: LET f=0
120 IF i$="c" THEN  LET x=0:
LET y=0: LET c(ln)=x: LET d(ln
)=y
130 IF i$="m" THEN  LET a(ln)=
x: LET b(ln)=y
140 IF i$="i" THEN  GOSUB 420
150 IF i$="o" THEN  GOSUB 600
160 IF i$="d" THEN  GOSUB 710
170 IF i$="z" THEN  LET lv=0:
GOSUB 840
180 LET sp=256: IF CODE INKEY$
<=41 THEN  LET sp=2048
190 IF INKEY$="8" OR INKEY$="(
" AND x>-32768+sp THEN  LET f=
1: LET x=x+sp: LET c(ln)=c(ln)
+sp
200 IF INKEY$="5" OR INKEY$="%
" AND x<32678-sp THEN  LET f=-
1: LET x=x-sp: LET c(ln)=c(ln)
-sp
210 IF INKEY$="7" OR INKEY$="'
" AND y>-22400+sp THEN  LET f=
-1: LET y=y+sp: LET d(ln)=d(ln
)+sp
220 IF INKEY$="6" OR INKEY$="&
" AND y<22400-sp THEN  LET f=1
: LET y=y-sp: LET d(ln)=d(ln)-
sp
230 GOSUB 250
240 GOTO 100
250 PRINT AT 1,2; INVERSE 1;"
ESCALA:";: LET a$=STR$ sc: IF
LEN a$>4 THEN  LET a$=a$( TO 4
)
255 PRINT  INVERSE 1;a$;AT 1,
18;" LINHAS:";ln-1
280 LET bx=FN a(a(ln)): LET by
=FN b(b(ln)): LET ex=FN a(c(ln
)): LET ey=FN b(d(ln))
290 PLOT bx,by: PLOT ex,ey
310 PLOT  OVER 1;bx,by: PLOT
OVER 1;ex,ey
320 RETURN
420 CLS
430 INPUT "NOME DO ARQUIVO ?";
f$
435 IF f$="" THEN  GOTO 430
440 LET ln=1: LET zoom=1: LET
sc=1
460 LOAD f$ DATA a()
470 LOAD f$ DATA b()
480 LOAD f$ DATA c()
485 LOAD f$ DATA d()
490 LET ln=a(601)
590 RETURN
600 CLS
610 INPUT "NOME DO ARQUIVO ?";
f$
615 IF f$="" THEN  GOTO 610
620 LET lv=1: GOSUB 840
625 LET a(601)=ln
630 SAVE f$ DATA a()
640 SAVE f$ DATA b()
650 SAVE f$ DATA c()
660 SAVE f$ DATA d()
700 RETURN
710 INPUT "NUMERO DE LINHAS A
APAGAR ?"'k
750 IF k=0 OR ln-k<=0 THEN
GOTO 830
755 LET ln=ln-k
760 LET x=a(ln): LET y=b(ln)
780 CLS
790 LET c(ln)=x: LET d(ln)=y
800 IF ln=1 THEN  GOTO 830
```

```
810 IF ABS (c(ln-1))<32767 AND
ABS (d(ln-1))<22399 THEN  LET
x=0: LET y=0
820 LET lv=2: GOSUB 840
830 RETURN
840 IF ln=0 THEN  RETURN
850 IF lv=1 THEN  LET zoom=1/
sc: CLS : GOTO 920
860 IF lv=2 THEN  LET zoom=1:
GOTO 920
870 SOUND .1,10: INPUT "DIGITE
A ESCALA  -  ZOOM ";zoom
890 IF zoom=0 THEN  GOTO 1020
910 CLS
920 FOR u=1 TO ln-1
930 LET a(u)=(a(u)-x)*zoom:
LET b(u)=(b(u)-y)*zoom
940 LET c(u)=(c(u)-x)*zoom:
LET d(u)=(d(u)-y)*zoom
950 IF ABS (a(u))<32768 AND
ABS (b(u))<22400 AND ABS (c(u)
)<32768 AND ABS (d(u))<22400
THEN  PLOT FN a(a(u)),FN b(b(u
)): DRAW FN a(c(u))-PEEK 23677
,FN b(d(u))-PEEK 23678
960 NEXT u
970 LET a(u)=(a(u)-x)*zoom:
LET b(u)=(b(u)-y)*zoom
980 LET c(u)=(c(u)-x)*zoom:
LET d(u)=(d(u)-y)*zoom
990 LET x=c(ln): LET y=d(ln)
1000 IF ABS (a(ln))>32767 OR AB
S (b(ln))>22399 THEN  LET a(ln)
=x: LET b(ln)=y
1010 LET sc=sc*zoom
1030 RETURN
```



```
10 PMODE 4,1:COLOR 0,1:PCLS:SCR
EEN 1,0:V=247
20 DIM NU$(10):FOR I=0 TO 10:RE
AD NU$(I):NEXT
30 DEF FN A(X)=(X/256)+128
40 DEF FN B(X)=(X/256)+96
50 LN=0:SC=1:L=600
60 X=0:Y=0:ZOOM=1
70 DIM BX(L),BY(L),EX(L),EY(L)
80 BX(0)=X:BY(0)=Y
90 EX(0)=X:EY(0)=Y
100 I$=INKEY$
110 IF I$="L" AND F AND L>LN TH
EN LINE(FNA(BX(LN)),FNB(BY(LN))
)-(FNA(EX(LN)),FNB(EY(LN))),PSE
T:LN=LN+1:BX(LN)=X:BY(LN)=Y:EX(
LN)=X:EY(LN)=Y:F=0
120 IF I$="C" THEN X=0:Y=0:EX(L
N)=X:EY(LN)=Y
```

```
130 IF I$="M" THEN BX(LN)=X:BY(
LN)=Y
140 IF I$="I" GOSUB 420
150 IF I$="O" GOSUB 600
160 IF I$="D" GOSUB 710
170 IF I$="Z" THEN LV=0:GOSUB 8
40
180 IF PEEK(339)=191 THEN SP=20
48 ELSE SP=256
190 IF PEEK(343)=V AND X>-32768
+SP THEN F=-1:X=X-SP:EX(LN)=EX(
LN)-SP
200 IF PEEK(344)=V AND X<32768-
SP THEN F=-1:X=X+SP:EX(LN)=EX(L
N)+SP
210 IF PEEK(341)=V AND Y>-24576
+SP THEN F=-1:Y=Y-SP:EY(LN)=EY(
LN)-SP
220 IF PEEK(342)=V AND Y<24576-
SP THEN F=-1:Y=Y+SP:EY(LN)=EYT(
LN)+SP
230 GOSUB 250
240 GOTO 100
250 SS$=STR$(SC)
260 DRAW"BM0,183":GOSUB 330
270 SS$=STR$(LN):DRAW"BM200,183
":GOSUB 330
280 BX=FNA(BX(LN)):BY=FNB(BY(LN
)):EX=FNA(EX(LN)):EY=FNB(EY(LN)
)
290 PSET(BX,BY):PSET (EX,EY)
300 FOR D=1 TO 50:NEXT D
310 PRESET(BX,BY):PRESET(EX,EY)
320 RETURN
330 FOR I=1 TO LEN(SS$)
340 D1=ASC(MID$(SS$,I,1))-48
350 IF DI=-2 THEN DI=10
360 IF DI<0 OR DI>10 THEN 380
370 DRAW"C1;XNU$(8);C0;BL8"+NU$
(DI)+"BR2"
380 NEXT I
390 RETURN
400 DATA R6D8L6U8BR8,BR6ND8BR2,
R6D4L6D4R6BR2BU8,R6D4NL3D4NL6BR
2BU8,D4R6D4U8BR2,NR6D4R6D4L6BE8
410 DATA D8R6U4L6U4BR8,R6ND8BR2
,R6D8L6U8D4R6U4BR2,D4R6D4U8L6BR
8,BR3BD8NR1BR5BU8
420 CLS
430 PRINT @256,"";:LINE INPUT"N
OME DO ARQUIVO A CARREGAR  ?";F
$
440 LN=0:ZOOM=1:SC=1
450 PCLS:SCREEN 1,0
460 OPEN"I",#-1,F$
470 INPUT#-1,N$
480 LN=VAL(N$)
490 FOR U=0 TO LN-1
500 INPUT #-1,BX$,BY$,EX$,EY$
510 BX(U)=VAL(BX$):BY(U)=VAL(BY
$):EX(U)=VAL(EX$):EY(U)=VAL(EY$
)
520 IF ABS(BX(U))<32768 AND ABS
(BY(U))<24576 AND ABS(EX(U))<32
768 AND ABS(EY(U))<24576 THEN L
INE (FNA(BX(U)),FNB(BY(U)))-(FN
A(EX(U)),FNB(EY(U))),PSET
530 NEXT
540 INPUT #-1,BX$,BY$,EX$,EY$
550 BX(U)=VAL(BX$):BY(U)=VAL(BY
$):EX(U)=VAL(EX$):EY(U)=VAL(EY$
)
560 X=EX(LN):Y=EY(LN)
570 CLOSE #-1
580 SCREEN 1,0
590 RETURN
600 CLS
610 PRINT @256,"";:LINE INPUT "
NOME DO ARQUIVO A SALVAR  ?";F$
620 LV=1:GOSUB 840
630 OPEN "O",#-1,F$
640 PRINT#-1,STR$(LN)
650 FOR U=0 TO LN
660 PRINT #-1,STR$(BX(U)),STR$(
BY(U)),STR$(EX(U)),STR$(EY(U))
679 NEXT
```

```
680 CLOSE #-1
690 SCREEN 1,0
700 RETURN
710 CLS
720 PRINT @256,"QUANTAS LINHAS
QUER APAGAR ";
730 INPUT K
740 SCREEN 1,0
750 IF K=0 OR LN-K<0 THEN 830
760 LN=LN-K
770 X=BX(LN):Y=BY(LN)
780 PCLS
790 EX(LN)=X:EY(LN)=Y
800 IF LN=0 THEN 830
810 IF ABS(EX(LN-1))<32767 AND
ABS(EY(LN-1))<24576 THEN X=0:Y=
0
820 LV=2:GOSUB 840
830 SCREEN 1,0:RETURN
840 IF LN=0 THEN RETURN
850 IF LV=1 THEN ZOOM=1/SC:PCLS
:GOTO 920
860 IF LV=2 THEN ZOOM=1:GOTO 92
0
870 CLS
880 PRINT @256," DIGITE A ESCAL
A / ZOOM ";
890 INPUT ZOOM
900 IF ZOOM=0 THEN 1020
910 PCLS:SCREEN 1,0
920 FOR U=0 TO LN-1
930 BX(U)=(BX(U)-X)*ZOOM:BY(U)=
(BY(U)-Y)*ZOOM
940 EX(U)=(EX(U)-X)*ZOOM:EY(U)=
(EY(U)-Y)*ZOOM
950 IF ABS(BX(U))<32768 AND ABS
 (BY(U))<24576 AND ABS(EX(U))<3
2768 AND ABS(EY(U))<24576 THEN
LINE(FNA(BX(U)),FNB(BY(U)))-(FN
A(EX(U)),FNB(EY(U))),PSET
960 NEXT
970 BX(U)=(BX(U)-X)*ZOMM:BY(U)=
(BY(U)-Y)*ZOOM
980 EX(U)=(EX(U)-X)*ZOOM:EY(U)=
(EY(U)-Y)*ZOOM
990 X=EX(LN):Y=EY(LN)
1000 IF ABS(BX(LN))>32767 OR AB
S(BY(LN))>24576 THEN BX(LN)=X:B
Y(LN)=Y
1010 SC=SC*ZOOM
1020 SCREEN 1,0
1030 RETURN
```

```
5  LOMEM: 16384
10  HGR : HCOLOR= 3: HOME :D$ =
  CHR$ (13) +  CHR$ (4)
30  DEF  FN A(X) = (X / 256) +
128
40  DEF  FN B(X) = (X / 256) +
96
50 LN = 0:SC = 1:L = 600:LM = 8
192
60 X = 0:Y = 0:ZOOM = 1:SP = 25
6
70  DIM BX(L),BY(L),EX(L),EY(L)

80 BX(0) = X:BY(0) = Y
90 EX(0) = X:EY(0) = Y
95  GOSUB 240
100  GET I$
110  IF I$ = "L" AND F AND L >
LN THEN  HPLOT  FN A(BX(LN)), F
N B(BY(LN)) TO  FN A(EX(LN)), F
N B(EY(LN)): VTAB 21:LN = LN +
1:BX(LN) = X:BY(LN) = Y:EX(LN)
= X:EY(LN) = Y:F = 0:M1 = 0:M2
= 0: GOSUB 320
120  IF I$ = "C" THEN X = 0:Y =
 0:EX(LN) = X:EY(LN) = Y
130  IF I$ = ";" THEN BX(LN) =
X:BY(LN) = Y
140  IF I$ = "E" THEN  GOSUB 42
0
```

```
150  IF I$ = "S" THEN  GOSUB 60
0
160  IF I$ = "D" THEN  GOSUB 71
0
170  IF I$ = "Z" THEN LV = 0: G
OSUB 840
180  IF I$ = "J" AND X >  - 327
68 THEN F =  - 1:X = X - SP:EX(
LN) = EX(LN) - SP
190  IF I$ = "K" AND X < 32768
 - SP THEN F =  - 1:X = X + SP:E
X(LN) = EX(LN) + SP
200  IF I$ = "I" AND Y >  - 245
76 + SP THEN F =  - 1:Y = Y - S
P:EY(LN) = EY(LN) - SP
210  IF I$ = "M" AND Y < 16128
THEN F =  - 1:Y = Y + SP:EY(LN)
 = EY(LN) + SP
220  GOSUB 240
230  GOTO 100
240  IF M <  > 0 THEN  POKE M,M
E: POKE M0,MA
245  VTAB 21: HTAB 1: CALL  - 9
58: PRINT "ESCALA:";SC,"LINHAS:
";LN
250 BX =  FN A(BX(LN)):BY =  FN
 B(BY(LN)):EX =  FN A(EX(LN)):E
Y =  FN B(EY(LN))
260 LX =  INT (EY / 64):CX =  I
NT (EY / 8) - (8 *  INT (EY / 6
4)):TX = EY - (8 *  INT (EY / 8
))
270 M = LM + (LX * 40) + (CX *
128) + (TX * 1024)
280 M = M + ( INT (EX / 7))
290 LX =  INT (BY / 64):CX =  I
NT (BY / 8) - (8 *  INT (BY / 6
4)):TX = BY - (8 *  INT (BY / 8
))
300 M0 = LM + (LX * 40) + (CX *
 128) + (TX * 1024)
310 M0 = M0 + ( INT (BX / 7))
320  IF M <  > M1 THEN M1 = M:M
E =  PEEK (M)
330  IF M0 <  > M2 THEN M2 = M0
:MA =  PEEK (M0)
340  HPLOT EX,EY: HPLOT BX,BY
350  RETURN
420  TEXT : HOME
430  INPUT "NOME DO ARQUIVO A S
ER LIDO:";F$
440 LN = 0:ZOOM = 1:SC = 1
450  HGR
460  PRINT D$;"OPEN ";F$
470  PRINT D$;"READ ";F$
480  INPUT LN
490  FOR U = 0 TO LN - 1
500  INPUT BX(U),BY(U),EX(U),EY
(U)
520  IF  ABS (BX(U)) < 32768 AN
D  ABS (BY(U)) < 24576 AND  ABS
 (EX(U)) < 32768 AND  ABS (EY(U
)) < 24576 THEN  HPLOT  FN A(BX
(U)), FN B(BY(U)) TO  FN A(EX(U
)), FN B(EY(U))
530  NEXT
540  INPUT BX(U),BY(U),EX(U),EY
(U)
560 X = EX(LN):Y = EY(LN)
570  PRINT D$;"CLOSE"
590  RETURN
600  TEXT : HOME
610  INPUT "NOME DO ARQUIVO A S
ER GRAVADO:";F$
620 LV = 1: GOSUB 840
630  PRINT D$;"OPEN ";F$
635  PRINT D$;"WRITE ";F$
640  PRINT LN
650  FOR U = 0 TO LN
660  PRINT BX(U): PRINT BY(U):
PRINT EX(U): PRINT EY(U)
670  NEXT
680  PRINT D$;"CLOSE"
700  RETURN
710  TEXT : HOME
720  INPUT "QUANTAS LINHAS QUER
 APAGAR? ";K
750  IF K = 0 OR LN - K < 0 THE
N 830
760 LN = LN - K
770 X = BX(LN):Y = BY(LN)
780  HGR
790 EX(LN) = X:EY(LN) = Y
800  IF LN = 0 THEN 830
810  IF  ABS (EX(LN - 1)) < 327
67 AND  ABS (EY(LN - 1)) < 2457
6 THEN X = 0:Y = 0
820 LV = 2: GOSUB 840
830  RETURN
840  IF LN = 0 THEN  RETURN
850  IF LV = 1 THEN ZOOM = 1 /
SC: GOTO 910
860  IF LV = 2 THEN ZOOM = 1: G
OTO 910
870  VTAB 23
880  INPUT "ESCALA DO ZOOM ? ";
ZOOM
900  IF ZOOM = 1 THEN LV = 1: G
OTO 850
905  IF ZOOM = 0 THEN ZOOM = 1
910  HGR
920  FOR U = 0 TO LN - 1
930 BX(U) = (BX(U) - X) * ZOOM:
BY(U) = (BY(U) - Y) * ZOOM
940 EX(U) = (EX(U) - X) * ZOOM:
EY(U) = (EY(U) - Y) * ZOOM
950  IF  ABS (BX(U)) < 32768 AN
D  ABS (BY(U)) < 24576 AND  ABS
 (EX(U)) < 32768 AND  ABS (EY(U
)) < 24576 THEN  HPLOT  FN A(BX
(U)), FN B(BY(U)) TO  FN A(EX(U
)), FN B(EY(U))
960  NEXT
970 BX(U) = (BX(U) - X) * ZOOM:
BY(U) = (BY(U) - Y) * ZOOM
980 EX(U) = (EX(U) - X) * ZOOM:
EY(U) = (EY(U) - Y) * ZOOM
990 X = EX(LN):Y = EY(LN)
1000  IF  ABS (BX(LN)) > 32767
OR  ABS (BY(LN)) > 24576 THEN B
X(LN) = X:BY(LN) = Y
1010 SC = SC * ZOOM
1030  RETURN
```

```
5 MAXFILES=2:CLEAR 2000
10 OPEN "GRP:" FOR OUTPUT AS #1
20 DEFFNA(X)=(X/256)+128
30 DEFFNB(X)=(X/256)+96
40 LN=0:SC=1:L=600:SP=256
50 X=0:Y=0:ZOOM=1
60 DIM BX(L),BY(L),EX(L),EY(L)
70 BX(0)=X:BY(0)=Y
80 EX(0)=X:BY(0)=Y
90 GOSUB 1200
100 I$=INKEY$
110 IF I$="L" AND F AND L>LN TH
EN LINE (FNA(BX(LN)),FNB(BY(LN)
))-(FNA(EX(LN)),FNB(EY(LN))),15
:LN=LN+1:BX(LN)=X:BY(LN)=Y:EX(L
N)=X:EY(LN)=Y:F=0
120 IF I$="C" THEN X=0:Y=0:EX(L
N)=X:EY(LN)=Y
130 IF I$="M" THEN BX(LN)=X:BY(
LN)=Y
140 IF I$="E" THEN GOSUB 420
150 IF I$="S" THEN GOSUB 600
160 IF I$="D" THEN GOSUB 710
170 IF I$="Z" THEN LV=0:GOSUB 8
40
```

```
180 IF I$=CHR$(29) AND X > -327
68! + SP THEN F=-1:X=X-SP:EX(LN
)=EX(LN)-SP
190 IF I$=CHR$(28) AND X < 3276
8! -SP THEN F=-1:X=X+SP:EX(LN)=
EX(LN)+SP
200 IF I$=CHR$(30) AND Y>-24576
+SP THEN F=-1:Y=Y-SP:EY(LN)=EY(
LN)-SP
```

```
210 IF I$=CHR$(31) AND Y<24576-
SP THEN F=-1:Y=Y+SP:EY(LN)=EY(L
N)+SP
220 GOSUB 250
230 GOTO 100
250 BX=FNA(BX(LN)):BY=FNB(BY(LN
)):EX=FNA(EX(LN)):EY=FNB(EY(LN)
)
260 PUT SPRITE 1,(BX-4,BY-4),15
,1
270 PUT SPRITE 2,(EX-4,EY-4),15
,1
280 LINE (10,180)-(255,191),4,B
F
290 PRESET (10,180):PRINT#1,"ES
C:";SC;"    LIN:";LN
300 RETURN
420 SCREEN0
430 INPUT"NOME DE ARQUIVO A SER
 CARREGADO:";F$
440 LN=0:ZOOM=1:SC=1:GOSUB 1200
450 F$="CAS:"+F$
460 OPEN F$ FOR INPUT AS #2
470 INPUT #2,LN
490 FOR U=0 TO LN-1
500 INPUT #2,BX,BY,EX,EY
510 BX(U)=BX:BY(U)=BY:EX(U)=EX:
EY(U)=EY
520 IF ABS(BX(U))<32768! AND AB
S(BY(U))<24576 AND ABS(EX(U))<3
2768! AND ABS(EY(U))<24576 THEN
 LINE(FNA(BX(U)),FNB(BY(U)))-(F
NA(EX(U)),FNB(EY(U))),15
530 NEXT
540 INPUT#2, BX,BY,EX,EY
550 BX(U)=BX:BY(U)=BY:EX(U)=EX:
EY(U)=EY
560 X=EX(LN):Y=EY(LN)
570 CLOSE #2
590 RETURN
600 SCREEN0
610 INPUT"NOME DE ARQUIVO PARA
GRAVAR:";F$:F$="CAS:"+F$
620 LV=1:GOSUB 1200:GOSUB 840
630 OPEN F$ FOR OUTPUT AS #2
640 PRINT#2,LN
650 FOR U=0 TO LN
660 PRINT#2,BX(U),BY(U),EX(U),E
Y(U)
670 NEXT
680 CLOSE #2
700 RETURN
710 SCREEN0
720 INPUT"QUER APAGAR QUANTAS L
INHAS";K
730 GOSUB 1200
750 IF K=0 OR LN-K<0 THEN 820
760 LN=LN-K
770 X=BX(LN):Y=BY(LN)
790 EX(LN)=X:EY(LN)=Y
800 IF LN=0 THEN 830
810 IF ABS(EX(LN-1))<32767 AND
ABS(EY(LN-1))<24576 THEN X=0:Y=
0
820 LV=2:GOSUB 840
830 RETURN
840 IF LN=0 THEN RETURN
850 IF LV=1 THEN ZOOM=1/SC:GOTO
 920
860 IF LV=2 THEN ZOOM=1:GOTO 92
0
870 SCREEN0
```

### IDÉIAS PARA O JOGO

Quando você usar o programa apresentado neste artigo como um jogo, a tarefa mais difícil será fazer com que o "tesouro" oculto na figura não apareça como uma mancha muito óbvia para quem o está procurando.

Existem, para isso, vários truques. Um deles consiste em executar um desenho bem elaborado e ocultar o tesouro em um dos detalhes. A dificuldade será ainda maior se você desenhar os detalhes em diferentes graus de ampliação. Isso obrigará seu oponente a também utilizar vários graus de ampliação na procura do tesouro.

Outro truque é a inclusão, no desenho, de pequenos detalhes parecidos com o tesouro, quando se utilizam as reduções. Seu oponente terá que investigar um a um, perdendo precioso tempo com isso.

A diversão será maior se você adotar alguma forma de contagem de pontos. Verifique, por exemplo, o número de vezes que o jogador muda a escala de aumento, ou o número de movimentações que ele precisa fazer até achar o tesouro. Outra alternativa é medir o tempo transcorrido entre o início e o fim da busca.

```
880 INPUT "ESCALA DO ZOOM:";ZOO
M
890 GOSUB 1200
900 IF ZOOM=0 THEN 1030
920 FOR U=0 TO LN-1
930 BX(U)=(BX(U)-X)*ZOOM:BY(U)=
(BY(U)-Y)*ZOOM
940 EX(U)=(EX(U)-X)*ZOOM:EY(U)=
(EY(U)-Y)*ZOOM
950 IF ABS(BX(U))<32768! AND AB
S(BY(U))<24576 AND ABS(EX(U))<3
2768! AND ABS(EY(U))<24576 THEN
 LINE(FNA(BX(U)),FNB(BY(U)))-(F
NA(EX(U)),FNB(EY(U))),15
960 NEXT
970 BX(U)=(BX(U)-X)*ZOOM:BY(U)=
(BY(U)-Y)*ZOOM
980 EX(U)=(EX(U)-X)*ZOOM:EY(U)=
(EY(U)-Y)*ZOOM
990 X=EX(LN):Y=EY(LN)
1000 IF ABS(BX(LN))>32767 OR AB
S(BY(LN))>24576 THEN BX(LN)=X:B
Y(LN)=Y
1010 SC=SC*ZOOM
1030 RETURN
1100 DATA 0,0,16,56,16,0,0,0
1200 COLOR 15,4,4:SCREEN 2,0:RE
STORE
1210 FOR I=1 TO 8:READ S:S$=S$+
CHR$(S):NEXT:SPRITE$(1)=S$
1220 RETURN
```

# PERIPÉCIAS NO REINO DE NETUNO

**Depois de viver mil aventuras em terra firme, Willie – o infortunado herói do jogo Avalanche – tem que se defrontar com os perigos de um mar agitado. Veja como ele se sai dessa.**

Willie está agora num morro a beira-mar. Mas não tem tempo de admirar a paisagem: se ele perder a calma quando as pedras estiverem rolando morro abaixo, ou se tremer com o barulho das cobras, descobrirá tardiamente que está em vias de afogar-se. Para criar o cenário adequado, devemos providenciar vários baldes de água do mar.

A rotina seguinte dá início ao avanço da maré; logo, é melhor que Willie não perca tempo no caminho e retorne rapidamente a seu gostoso piquenique.

```
 10 REM org 58882
 20 REM mea ld bc,57312
 30 REM ld a,(57353)
 40 REM bit 2,a
 50 REM jr z,mpt
 60 REM ld bc,57320
 70 REM mpt ld hl,(57354)
 80 REM ld a,15
 90 REM ld d,32
100 REM mpu push de
110 REM push bc
120 REM call print
130 REM inc hl
140 REM pop bc
150 REM pop de
160 REM dec d
170 REM jr nz,mpu
180 REM ld a,(57353)
190 REM dec a
200 REM ld (57353),a
210 REM jr nz,mrt
220 REM ld a,10
230 REM ld (57353),a
240 REM ld hl,(57354)
250 REM ld de,32
260 REM sbc hl,de
270 REM ld (57354),hl
280 REM mrt ret
290 REM org 58217
300 REM print *
```

Embora pareça que temos na tela um trecho do mar, há na verdade apenas dois caracteres. O primeiro ocupa as oito posições do endereço 57312 em diante, e o segundo as oito posições a partir do endereço 57320. Você pode achar que isso não é água suficiente nem mesmo para molhar os pés de nosso herói. Mas, quando esses dois caracteres forem impressos consecutivamente em linhas alternadas, você construirá rapidamente um oceano.

Como o mar será impresso na tela na proporção de um caractere por vez, recorreremos à rotina **print** de novo. Assim, os parâmetros necessários devem ser carregados nos registros corretos. Como de costume, o par BC carrega o apontador do primeiro byte de dados. O acumulador A carrega o código da cor e o par HL contém a posição da tela onde o dado será impresso. Depois desse lembrete, carregamos o par de registros BC com o endereço do byte inicial que forma o primeiro caractere de mar.

A variável no endereço 57353 é chamada variável de atraso do mar; é ela que controla o movimento das águas. O bit 2 é usado como uma baliza para informar ao processador qual dos dois caracteres de mar deverá ser impresso na última linha.

O conteúdo do atraso do mar é colocado no acumulador e a instrução **bit 2,a** analisa esse bit em particular. Se o seu valor for 0, a instrução **jr z** que vem em seguida faz o processador saltar para a próxima instrução. Mas, se o valor do bit 2 é 1, o salto não ocorre e o par BC é carregado com o endereço do padrão inicial do segundo caractere de mar.

## MAR AGITADO

O par de registros HL é carregado com o conteúdo das posições de memória 57354 e 57355. Esses endereços contêm a posição do mar que está sendo impressa no momento e que foi inicializada pela rotina da parte sete de *Avalanche* com a posição do canto inferior esquerdo da tela.

O acumulador A é carregado com o número 15 para dar ao mar a tonalidade azul adequada. O registro D, por sua vez, é carregado com o número 32; ele será usado como contador para somar 32 colunas através da tela.

O contador em D deve ser preservado intacto. Se o processador utilizar o registro D na rotina **print**, ele se perderá; assim, o par DE é guardado na pilha. Os dados para o caractere de mar serão usados outras vezes, já que cada linha de mar é formada pelo mesmo caractere impresso 32 vezes. Dessa forma, o par BC é guardado na pilha também. A rotina **print** é chamada e os oito bytes do caractere de mar são impressos no lugar correto da tela.

O par HL é então incrementado, movendo o apontador de tela para a próxima posição ao longo da linha. Ao mesmo tempo, o apontador de dados é

- O AVANÇO DA MARÉ
- IMPRESSÃO DOS CARACTERES DO MAR
- TONALIDADE DO MAR
- VARIÁVEL DE ATRASO
- REPETIÇÃO DOS CARACTERES EM LINHAS ALTERNADAS
- A ROTINA **CHARPR**
- MAR BRAVIO

levado de volta para o início dos padrões do caractere (ele tinha sido incrementado durante a rotina **print**), sendo recuperado da pilha. O contador de colunas é também recuperado da pilha e decrementado. Se ele não foi reduzido a zero, a instrução **jr nz** retornará ao início do laço e o processador imprimirá o caractere de mar na posição seguinte da tela ao longo dessa linha.

Quando o contador em D tiver sido reduzido a zero, o processador saltará para fora do laço e prosseguirá com a próxima instrução.

## TEMPO E MARÉ

O atraso do mar é carregado sobre o acumulador, decrementado e armazenado de volta em 57353. Se ele não tiver sido reduzido a 0, a instrução **jr nz** fará o processador saltar as instruções restantes até o final da rotina para a qual ele retornará. Se ele tiver sido reduzido a zero, o atraso do mar será ajustado para 10.

O apontador de posição do mar é carregado em HL; 32 é carregado em DE e subtraído de HL; o resultado é armazenado de volta no apontador de localização do mar em 57354. Assim, na próxima vez que a rotina for chamada, o mar terá subido uma linha.

O programa a seguir inicia a expansão da maré.

```
10   ORG 19678
20   SEA LDU #18206
30   LDA 18246
40   BITA #2
50   BEQ SPT
60   LDU #18222
70   SPT LDX 18247
80   LDA #16
90   SPTI PSHS A,U
100   JSR CHARPR
110   PULS U,A
120   DECA
130   BNE SPTI
140   DEC 18246
150   BNE SRT
160   LDA #10
170   STA 18246
180   LDX 18247
190   LEAX -256,X
200   STX 18247
210   SRT RTS
220   CHARPR EQU 19402
```

Para testar esse programa, você precisa carregá-lo no resto de *Avalanche* e executar a seguinte rotina:

```
5 POKE &H467F,&H4C:POKE &H4C80,
&HF3
10 EXEC 19426
20 FOR G=1 TO 160
30 EXEC 19678
40 FOR H=1 TO 100:NEXT H,G
50 GOTO 50
```

Quando esta rotina estiver funcionando, a maré subirá até que a tela esteja completamente cheia de água. Isso nunca aconteceria durante o jogo, pois Willie teria se afogado antes e o jogo voltaria ao início.

## PEQUENAS GOTAS DE ÁGUA

Embora aparentemente tenhamos um trecho do mar representado na tela, existem na verdade apenas dois caracteres importantes na tabela de dados. O primeiro desses caracteres ocupa os oito endereços a partir de 18206 e o segundo ocupa os oito a partir de 18222. Quando esses dois caracteres forem impressos repetidamente em linhas alternadas, você terá um oceano.

Como o mar será impresso na tela do microcomputador, segundo um ritmo de um caractere de cada vez, a rotina **CHARPR** será usada de novo. Logo, seus parâmetros têm de ser carregados nos registros corretos. Como sempre, o registro U carrega o apontador do primeiro byte de dados; então, a própria tabela de dados atua como pilha do usuário. O registro X, por sua vez, car-

rega a posição da tela onde o dado será impresso. U é então carregado com o endereço do primeiro byte que forma o primeiro caractere de mar.

A variável no endereço 18246 é chamada variável de atraso do mar e controla o movimento da maré. O bit 2 dessa variável é usado como uma baliza para informar ao processador qual caractere de mar será usado na linha que está sendo impressa.

O conteúdo do atraso do mar é carregado no acumulador e a instrução **BITA #2** analisa esse bit em particular. Se o valor do bit não for 1, a instrução **BEQ** fará o processador pular a próxima instrução através do rótulo **SPT**. Mas, se o valor for 1, o salto não ocorrerá e U será carregado com o endereço do primeiro byte que forma o segundo caractere de mar.

## PRIMEIRO CARACTERE

O registro X é carregado com o conteúdo da posição de memória 18247. Esse endereço carrega a posição em que o mar está sendo impresso no momento em que foi inicializada pela rotina da parte sete de *Avalanche* no canto inferior esquerdo da tela. O acumulador A é carregado com 16, e será usado como contador para fazer com que dezesseis caracteres de mar sejam impressos.

Esse contador precisa ser preservado intacto enquanto o processador executa a rotina **CHARPR**; portanto, guardamos A na pilha do usuário. Os dados para o caractere de mar serão usados várias vezes, porque cada linha do mar é feita do mesmo caractere — assim, o registro U também é guardado na pilha. A rotina **CHARPR** é chamada e os oito bytes do caractere de mar são impressos no lugar correto da tela.

## VEJA O MAR

A rotina **CHARPR** incrementa automaticamente o registro X, de modo que ele esteja pronto para imprimir o próximo caractere ao longo da linha. O apontador de dados é movido de volta ao início dos bytes que formam o caractere de mar por meio de sua recuperação da pilha; esse apontador foi incrementado durante a rotina **CHARPR**. O contador também é recuperado da pilha.

O contador é então decrementado e, se ainda não for zero, a instrução **BNE** saltará para o laço onde o processador imprime o caractere de mar na próxima posição ao longo da linha atual.

Quando o contador em A já for zero, o processador sairá do laço e executará a instrução seguinte.

## MAR BRAVIO

O próximo passo consiste em decrementar o atraso do mar. Se esse atraso não tiver sido reduzido a zero, a instrução **BNE** fará o processador saltar as próximas instruções até o final da rotina, para onde ele retornará.

Se ele tiver sido reduzido a zero, o atraso do mar será reajustado com valor 10. O apontador de posição será então carregado no registro X e 256 subtraído dele. O resultado é armazenado

de volta nos endereços que guardam esse apontador, ou seja, 18247. Assim, da próxima vez que essa rotina for chamada, o mar subirá uma linha.

A rotina que vem a seguir inicia a expansão da maré; por isso, é melhor Willie andar o mais rápido possível e recuperar o seu lanche.

```
10    org 54218
20    sea ld b,72
30    ld a,(-5213)
40    bit 2,a
50    jr z,spt
60    ld b,76
70    spt ld hl,(62407)
80    ld de,(-5212)
90    add hl,de
100    ld a,b
110    ld bc,32
120    call 86
130    ld a,(-5213)
140    dec a
150    ld (-5213),a
160    jr nz,srt
170    ld a,10
180    ld (-5213),a
190    ld hl,(-5212)
200    ld de,32
210    sbc hl,de
220    ld (-5212),hl
230    srt ret
240    end
```

Embora tenhamos a impressão de que há um trecho do mar na nossa tela, na verdade só existem dois padrões diferentes. O primeiro tem o código 72 na tabela de padrões e o segundo é o padrão de código 76. À primeira vista, a quantidade de água não é suficiente nem mesmo para molhar os pés de Willie. Mas, quando esses dois padrões forem impressos consecutivamente em linhas alternadas, em pouco tempo teremos, na tela, um oceano.

Como o mesmo padrão de mar será impresso 32 vezes na mesma linha, ocupando-a completamente, usaremos a rotina **86** da ROM. Essa rotina escreve um número na VRAM a partir de um endereço e um certo número de vezes.

Neste programa, a rotina 86 da ROM será usada para colocar o código de um padrão de mar várias vezes na tabela de nomes da VRAM; isso equivale a imprimir o mesmo caractere diversas vezes em seqüência, já que a tela é um reflexo da tabela de nomes.

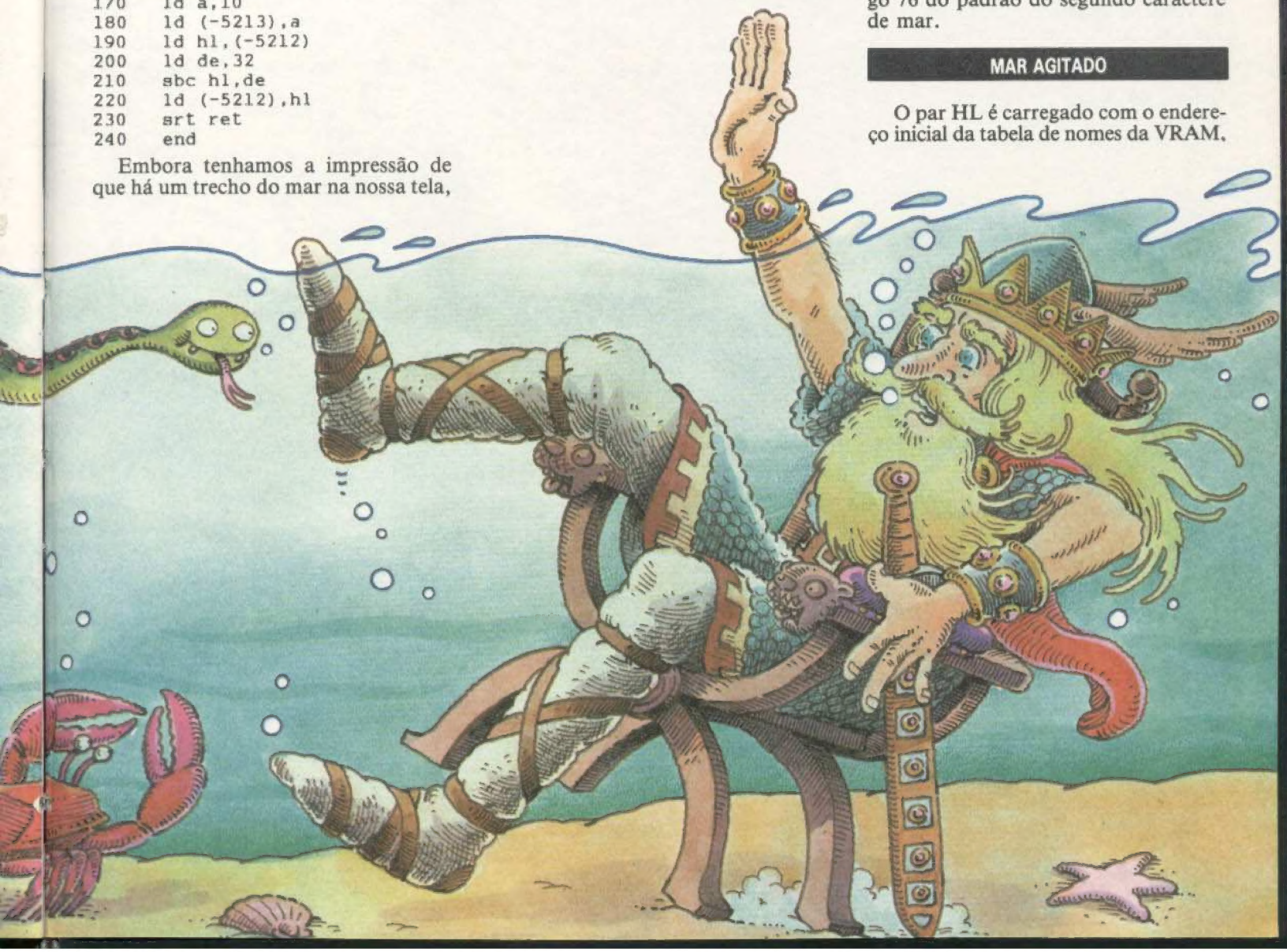
Os parâmetros necessários devem ser carregados nos registros corretos, antes de se chamar a rotina **86**. É preciso também que o acumulador A contenha o código do padrão que vai ser impresso. O par HL, por sua vez, contém a posição da tabela de nomes — ou seja, da tela, onde o padrão começará a ser impresso e o par de registros BC carrega o número de vezes que ele será impresso. Feito o lembrete, continuemos a examinar o programa. O código do padrão do primeiro caractere de mar é colocado no registro B.

A variável no endereço −5213 é chamada variável de atraso do mar; é ela que controla a subida da maré. O bit 2 dessa variável é usado como baliza para indicar ao processador qual dos dois caracteres de mar deve ser impresso na linha atual.

O conteúdo do atraso do mar é colocado no acumulador e a instrução **bit 2,a** analisa esse bit isoladamente. Se o seu valor for 0, a instrução **jr z** que vem em seguida fará o processador saltar a próxima instrução. Mas, se o valor do bit 2 for 1, o salto não será realizado e o registro B será carregado com o código 76 do padrão do segundo caractere de mar.

## MAR AGITADO

O par HL é carregado com o endereço inicial da tabela de nomes da VRAM,



que está armazenado nas posições 62407 e 62408. O par DE, por sua vez, é carregado com o valor contido nas posições −5212 e −5211; esse número corresponde à posição na tela do começo da linha que está sendo impressa atualmente e foi inicializado pela rotina da parte sete de *Avalanche* para indicar o canto inferior esquerdo da tela. Os conteúdos de HL e DE são somados em HL, que passa a abrigar o endereço correspondente na tabela de nomes.

O acumulador A recebe então o código do padrão que até agora estava em B. Ao mesmo tempo, o par BC é carregado com o número de vezes em que o padrão deve ser impresso na tela — ou seja, 32, que corresponde ao número de colunas na linha. A seguir, é chamada a rotina **86** da ROM; nesse momento, a linha de mar já estará impressa.

## TEMPO DE SUBIDA

O atraso do mar é carregado no acumulador, decrementado e armazenado de volta no endereço −5213. Se o seu valor ainda não tiver sido reduzido a zero, a instrução **jr nz** fará com que o processador salte as instruções restantes até o final da rotina, para a qual ele retornará.

Por outro lado, se ele tiver sido reduzido a zero, o atraso do mar será ajustado para 10.

O apontador de posição do mar é carregado em HL e o número 32 é subtraído dele por meio do par DE; o resultado é armazenado de volta no apontador de posição em −5212 e −5211. Assim, na próxima vez que a rotina for chamada, o mar terá subido uma linha.

Como você já deve ter notado, o atraso do mar controla o tempo que a rotina passará imprimindo a primeira linha; esgotado esse tempo, a rotina é ajustada para subir uma linha a cada dez chamadas.

Depois de jogar algumas vezes, talvez você queira reforçar a dose de emoção de *Avalanche*.

Torne a aventura mais difícil alternando a variável de atraso.

## CLASSIFICAÇÃO DOS MICROCOMPUTADORES POR LINHA

| LINHA | FABRICANTE | MODELO |
|---|---|---|
| Apple II + | Appletronica | Thor 2010 |
| Apple II + | CCE | MC-4000 Exato |
| Apple II + | CPA | Absolutus |
| Apple II + | CPA | Polaris |
| Apple II + | Digitus | DGT-AP |
| Apple II + | Dismac | D-8100 |
| Apple II + | ENIAC | ENIAC II |
| Apple II + | Franklin | Franklin |
| Apple II + | Houston | Houston AP |
| Apple II + | Magnex | DM II |
| Apple II + | Maxitronica | MX-2001 |
| Apple II + | Maxitronica | MX-48 |
| Apple II + | Maxitronica | MX-64 |
| Apple II + | Maxitronica | Maxitronic I |
| Apple II + | Microcraft | Craf II Plus |
| Apple II + | Milmar | Apple II Plus |
| Apple II + | Milmar | Apple Master |
| Apple II + | Milmar | Apple Senior |
| Apple II + | Omega | MC-400 |
| Apple II + | Polymax | Maxxi |
| Apple II + | Polymax | Poly Plus |
| Apple II + | Spectrum | Microengenho I |
| Apple II + | Spectrum | Spectrum ed |
| Apple II + | Suporte | Venus II |
| Apple II + | Sycomig | SIC I |
| Apple II + | Unitron | AP II |
| Apple II + | Victor do Brasil | Elppa II Plus |
| Apple II + | Victor do Brasil | Elppa Jr. |
| Apple IIe | Microcraft | Craft IIe |
| Apple IIe | Microdigital | TK-3000 IIe |
| Apple IIe | Spectrum | Microengenho II |
| MSX | Gradiente | Expert GPC-1 |
| MSX | Sharp | Hotbit HB-8000 |
| Sinclair Spectrum | Microdigital | TK-90X |
| Sinclair Spectrum | Timex | Timex 2000 |
| Sinclair ZX-81 | Apply | Apply 300 |
| Sinclair ZX-81 | Engebras | AS-1000 |
| Sinclair ZX-81 | Filcres | NEZ-8000 |
| Sinclair ZX-81 | Microdigital | TK-82C |
| Sinclair ZX-81 | Microdigital | TK-83 |
| Sinclair ZX-81 | Microdigital | TK-85 |
| Sinclair ZX-81 | Prologica | CP-200 |
| Sinclair ZX-81 | Ritas | Ringo R-470 |
| Sinclair ZX-81 | Timex | Timex 1000 |
| Sinclair ZX-81 | Timex | Timex 1500 |
| TRS-80 Mod. I | Dismac | D-8000 |
| TRS-80 Mod. I | Dismac | D-8001/2 |
| TRS-80 Mod. I | LNW | LNW-80 |
| TRS-80 Mod. I | Video Genie | Video Genie I |
| TRS-80 Mod.III | Digitus | DGT-100 |
| TRS-80 Mod.III | Digitus | DGT-1000 |
| TRS-80 Mod.III | Kemitron | Naja 800 |
| TRS-80 Mod.III | Prologica | CP-300 |
| TRS-80 Mod.III | Prologica | CP-500 |
| TRS-80 Mod.III | Sysdata | Sysdata III |
| TRS-80 Mod.III | Sysdata | Sysdata Jr. |
| TRS-80 Mod.IV | Multix | MX-Compacto |
| TRS-80 Mod.IV | Sysdata | Sysdata IV |
| TRS-Color | Codimex | CS-6508 |
| TRS-Color | Dynacom | MX-1600 |
| TRS-Color | LZ | Color 64 |
| TRS-Color | Microdigital | TKS-800 |
| TRS-Color | Prologica | CP-400 |

## CLASSIFICAÇÃO DOS MICROCOMPUTADORES POR NOME

| FABRICANTE | MODELO | PAÍS | LINHA |
|---|---|---|---|
| Appletronica | Thor 2010 | Brasil | Apple II + |
| Apply | Apply 300 | Brasil | Sinclair ZX-81 |
| CCE | MC-4000 Exato | Brasil | Apple II + |
| CPA | Absolutus | Brasil | Apple II + |
| CPA | Polaris | Brasil | Apple II + |
| Codimex | CS-6508 | Brasil | TRS-Color |
| Digitus | DGT-100 | Brasil | TRS-80 Mod.III |
| Digitus | DGT-1000 | Brasil | TRS-80 Mod.III |
| Digitus | DGT-AP | Brasil | Apple II + |
| Dismac | D-8000 | Brasil | TRS-80 Mod. I |
| Dismac | D-8001/2 | Brasil | TRS-80 Mod. I |
| Dismac | D-8100 | Brasil | Apple II + |
| Dynacom | MX-1600 | Brasil | TRS-Color |
| ENIAC | ENIAC II | Brasil | Apple II + |
| Engebras | AS-1000 | Brasil | Sinclair ZX-81 |
| Filcres | NEZ-8000 | Brasil | Sinclair ZX-81 |
| Franklin | Franklin | USA | Apple II + |
| Gradiente | Expert GPC1 | Brasil | MSX |
| Houston | Houston AP | Brasil | Apple II + |
| Kemitron | Naja 800 | Brasil | TRS-80 Mod.III |
| LNW | LNW-80 | USA | TRS-80 Mod. I |
| LZ | Color 64 | Brasil | TRS-Color |
| Magnex | DM II | Brasil | Apple II + |
| Maxitronica | MX-2001 | Brasil | Apple II + |
| Maxitronica | MX-48 | Brasil | Apple II + |
| Maxitronica | MX-64 | Brasil | Apple II + |
| Maxitronica | Maxitronic I | Brasil | Apple II + |
| Microcraft | Craft II Plus | Brasil | Apple II + |
| Microcraft | Caft IIe | Brasil | Apple IIe |
| Microdigital | TK-3000 IIe | Brasil | Apple IIe |
| Microdigital | TK-82C | Brasil | Sinclair ZX-81 |
| Microdigital | TK-83 | Brasil | Sinclair ZX-81 |
| Microdigital | TK-85 | Brasil | Sinclair ZX-81 |
| Microdigital | TK-90X | Brasil | Sinclair Spectrum |
| Microdigital | TKS-800 | Brasil | TRS-Color |
| Milmar | Apple II Plus | Brasil | Apple II + |
| Milmar | Apple Master | Brasil | Apple II + |
| Milmar | Apple Senior | Brasil | Apple II + |
| Multix | MX-Compacto | Brasil | TRS-80 Mod.IV |
| Omega | MC-400 | Brasil | Apple II + |
| Polymax | Maxxi | Brasil | Apple II + |
| Polymax | Poly Plus | Brasil | Apple II + |
| Prologica | CP-200 | Brasil | Sinclair ZX-81 |
| Prologica | CP-300 | Brasil | TRS-80 Mod.III |
| Prologica | CP-400 | Brasil | TRS-Color |
| Prologica | CP-500 | Brasil | TRS-80 Mod.III |
| Ritas | Ringo R-470 | Brasil | Sinclair ZX-81 |
| Sharp | Hotbit HB-8000 | Brasil | MSX |
| Spectrum | Microengenho I | Brasil | Apple II + |
| Spectrum | Microengenho II | Brasil | Apple IIe |
| Spectrum | Spectrum ed | Brasil | Apple II + |
| Suporte | Venus II | Brasil | Apple II + |
| Sycomig | SIC I | Brasil | Apple II + |
| Sysdata | Sysdata III | Brasil | TRS-80 Mod.III |
| Sysdata | Sysdata IV | Brasil | TRS-80 Mod.IV |
| Sysdata | Sysdata Jr. | Brasil | TRS-80 Mod.III |
| Timex | Timex 1000 | USA | Sinclair ZX-81 |
| Timex | Timex 1500 | USA | Sinclair ZX-81 |
| Timex | Timex 2000 | USA | Sinclair Spectrum |
| Unitron | AP II | Brasil | Apple II + |
| Victor do Brasil | Elppa II Plus | Brasil | Apple II + |
| Victor do Brasil | Elppa Jr. | Brasil | Apple II + |
| Video Genie | Video Genie I | USA | TRS-80 Mod. I |

## UM LOGOTIPO PARA CADA MODELO DE COMPUTADOR

**INPUT** foi especialmente projetado para microcomputadores compatíveis com as sete principais linhas existentes no mercado.
Os blocos de textos e listagens de programas aplicados apenas a determinadas linhas de micros podem ser identificados por meio dos seguintes símbolos:

Sinclair ZX-81

TRS-80

TK-2000

MSX

Spectrum

TRS-Color

Apple II

Quando o emblema for seguido de uma faixa, então tanto o texto como os programas que se seguem passam a ser específicos para a linha indicada.

# NOVOS LANÇAMENTOS, NOVOS SUCESSOS.

JÁ NAS LIVRARIAS

**A FRATERNIDADE DA PEDRA**
**David Morrell**

Um grupo secreto, sob direção de um padre armado, passa a agir contra o terrorismo. Mas, será que violência se combate com mais violência? Eis o dilema de Drew, agente da Lei envolvido com fatos e figuras do mundo real, num livro surpreendente do criador de Rambo.

**AVENTUREIROS E MILIONÁRIOS**
**Clark Howard**

No romance do Texas, a saga da descoberta de um poço de petróleo e o drama de um casal que herda um lote de terra aparentemente sem valor e enfrenta com coragem os poderosos do lugar, até vencer. Uma vitória antes de tudo moral, numa história forte e envolvente, com realistas cenas de amor.

**CAÇADA SEM FIM**
**Bryan Forbes**

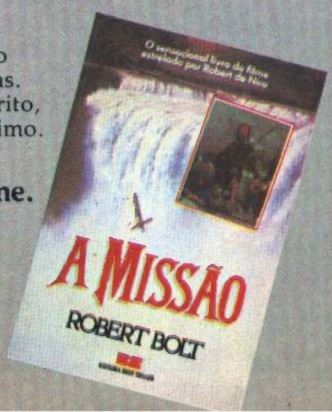
Uma brilhante história de espionagem envolvendo a KGB. Por que matar uma ex-espiã que já tinha sido desmascarada e torturada tempos atrás? Um agente inglês, seu antigo amante, enfrenta um desafio: descobrir por que ela foi morta... e por que *agora!*

**PODER**
**Howard Fast**

Um líder sindical com a volúpia do poder, a luta pelos direitos dos trabalhadores, nos Estados Unidos, e sua manipulação por corruptos e oportunistas; o jogo das ambições políticas. Admiravelmente escrito, um romance atualíssimo.

**SUPERSEXO**
**Alexandra Penney**

Se não for o primeiro, este vai ser o último e definitivo guia para o prazer que o leitor poderá seguir: um livro que derruba mitos, faz sugestões provocantes e propõe técnicas ousadas para se chegar ao supersexo, uma relação intensa e especial entre os casais, que não exclui o romantismo.




**Não perca também: A MISSÃO, de Robert Bolt, o livro do filme.**

EDITORA BEST SELLER